US009319690B2

(12) **United States Patent**
Balestri et al.

(10) **Patent No.: US 9,319,690 B2**
(45) **Date of Patent: Apr. 19, 2016**

(54) **METHOD AND SYSTEM FOR IMAGE ANALYSIS**

(71) Applicant: **TELECOM ITALIA S.p.A.**, Milan (IT)

(72) Inventors: **Massimo Balestri**, Turin (IT); **Gianluca Francini**, Turin (IT); **Skjalg Lepsoy**, Turin (IT)

(73) Assignee: **TELECOM ITALIA S.p.A.**, Milan (IT)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **14/396,525**

(22) PCT Filed: **Apr. 19, 2013**

(86) PCT No.: **PCT/EP2013/058205**
§ 371 (c)(1),
(2) Date: **Oct. 23, 2014**

(87) PCT Pub. No.: **WO2013/160210**
PCT Pub. Date: **Oct. 31, 2013**

(65) **Prior Publication Data**

US 2015/0104110 A1 Apr. 16, 2015

Related U.S. Application Data

(60) Provisional application No. 61/659,681, filed on Jun. 14, 2012.

(30) **Foreign Application Priority Data**

Apr. 23, 2012 (IT) MI2012A0675

(51) **Int. Cl.**
G06K 9/36 (2006.01)
H04N 19/124 (2014.01)
G06T 9/00 (2006.01)

(52) **U.S. Cl.**
CPC ***H04N 19/124*** (2014.11); ***G06T 9/00*** (2013.01)

(58) **Field of Classification Search**
CPC . H04N 7/26085; H04N 7/30; H04N 7/26244; H04N 7/50; H04N 7/26079; H04N 1/4052; H04N 1/4053; H04N 1/52; H04N 7/28; H04N 7/26313; G09G 3/2059; G06T 9/008; H03M 7/3082
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

2009/0304090 A1* 12/2009 Cordara H04N 19/537 375/240.26

OTHER PUBLICATIONS

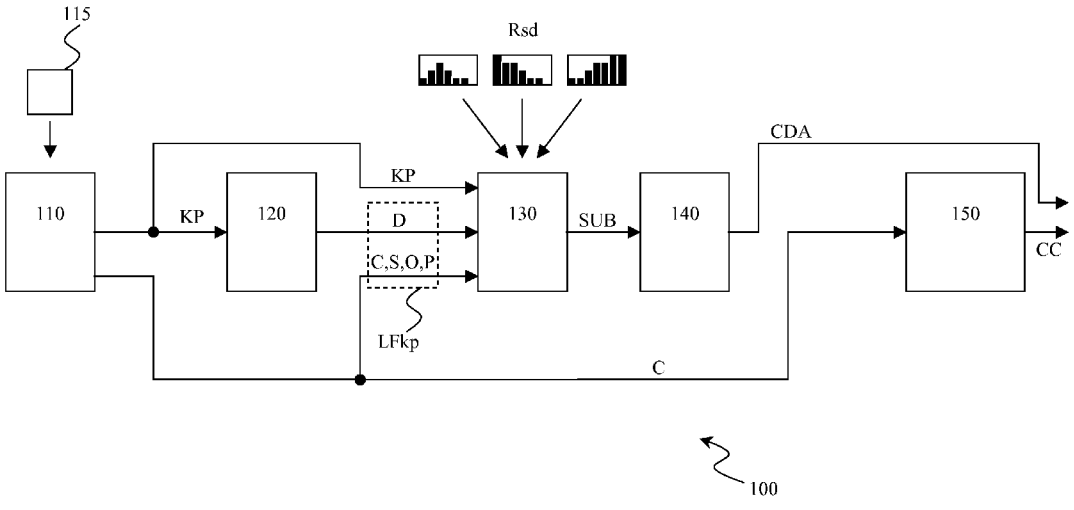
International Search Report issued Jun. 17, 2013, in PCT/EP13/058205 filed Apr. 19, 2013.
Written Opinion of the International Searching Authority issued Jun. 17, 2013, in PCT/EP13/058205 filed Apr. 19, 2013.
(Continued)

Primary Examiner — Duy M Dang
(74) *Attorney, Agent, or Firm* — Oblon, McClelland, Maier & Neustadt, L.L.P

(57) **ABSTRACT**

A method for processing an image includes: identifying a group of keypoints in the image; for each keypoint of the group; a) calculating a corresponding descriptor array including a plurality of array elements, each array element storing values taken by a corresponding color gradient histogram of a respective sub-region of the image in the neighborhood of the keypoint; b) generating at least one compressed descriptor array by compressing at least one portion of the descriptor array by vector quantization using a codebook including a plurality of codewords.

10 Claims, 27 Drawing Sheets



(56) **References Cited**

OTHER PUBLICATIONS

"Description of Test Model under Consideration for CDVS", Video Subgroup, International Organisation for Standardisation, Organisation Internationale de Normalisation, ISO/IEC JTC1/SC29/WG11 Coding of Moving Pictures and Audio, N12367, XP030018862, Dec. 2011, 24 pages.

Jegou, et al., "Product Quantization for Nearest Neighbor Search", IEEE Transactions on Pattern Analysis and Machine Intelligence, XP055026005, vol. 33, No. 1, Jan. 2011, pp. 117-128.

Chandrasekhar, et al., "Survey of SIFT Compression Schemes", The Second International Workshop on Mobile Multimedia Processing—in Conjunction with the 20th International Conference on Pattern Recognition (ICPR 2010), XP-002631251, Aug. 2010, 8 pages.

* cited by examiner

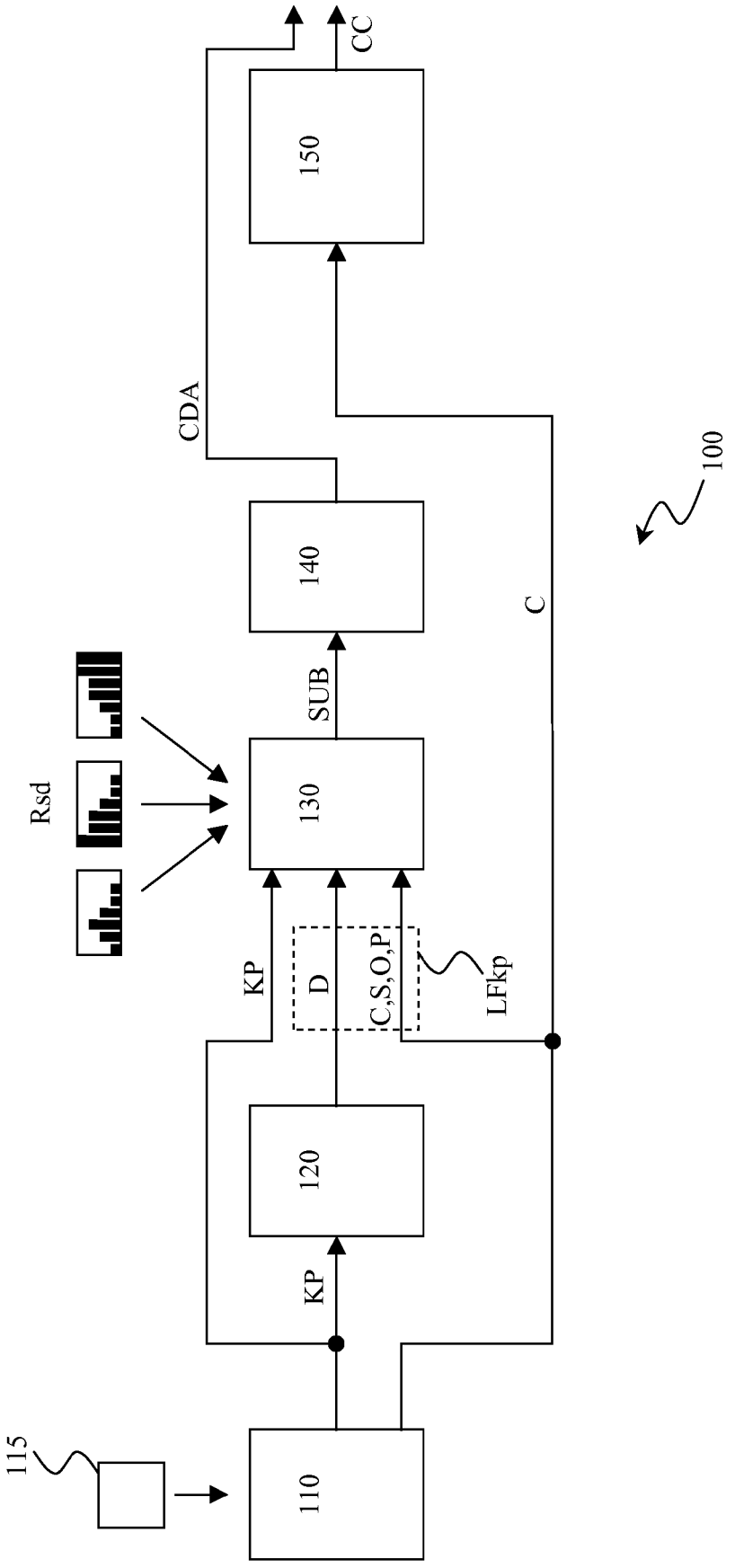


FIG.1

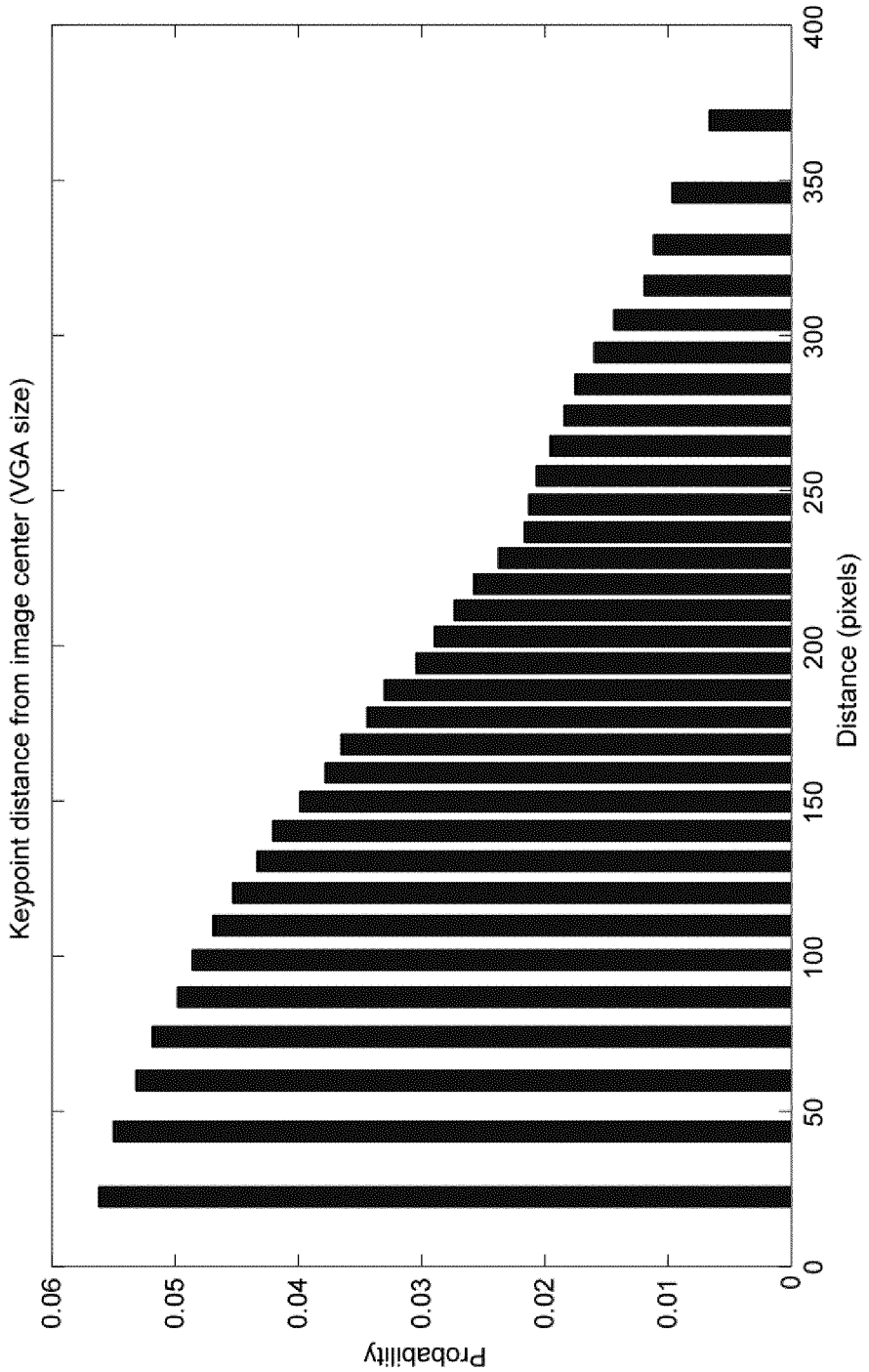

Keypoint distance from image center (VGA size)
Probability
Distance (pixels)
0.06
0.05
0.04
0.03
0.02
0.01
0
50
100
150
200
250
300
350
400

FIG.2A

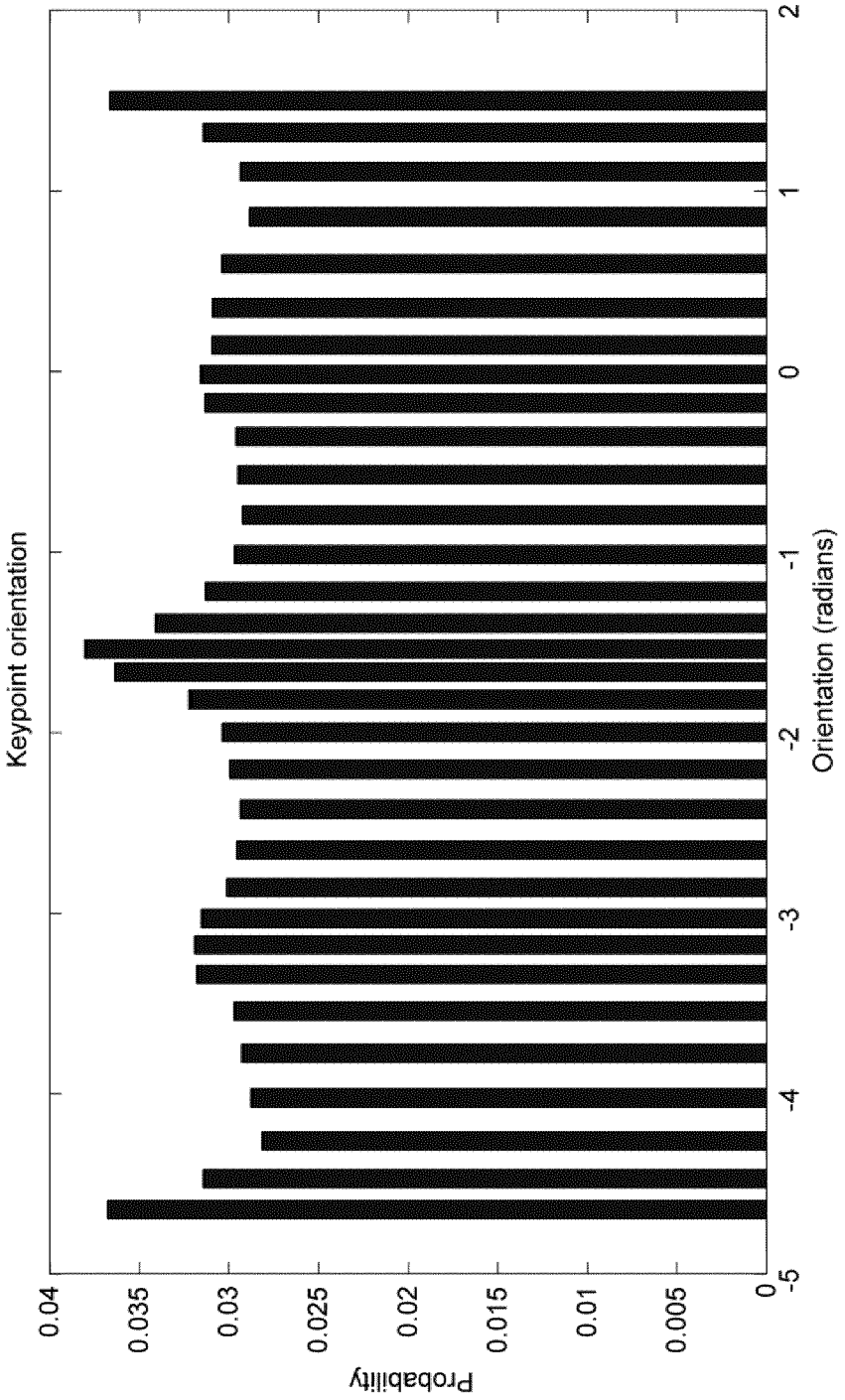

Keypoint orientation
Probability
0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
-5
-4
-3
-2
-1
0
1
2
Orientation (radians)

FIG.2B

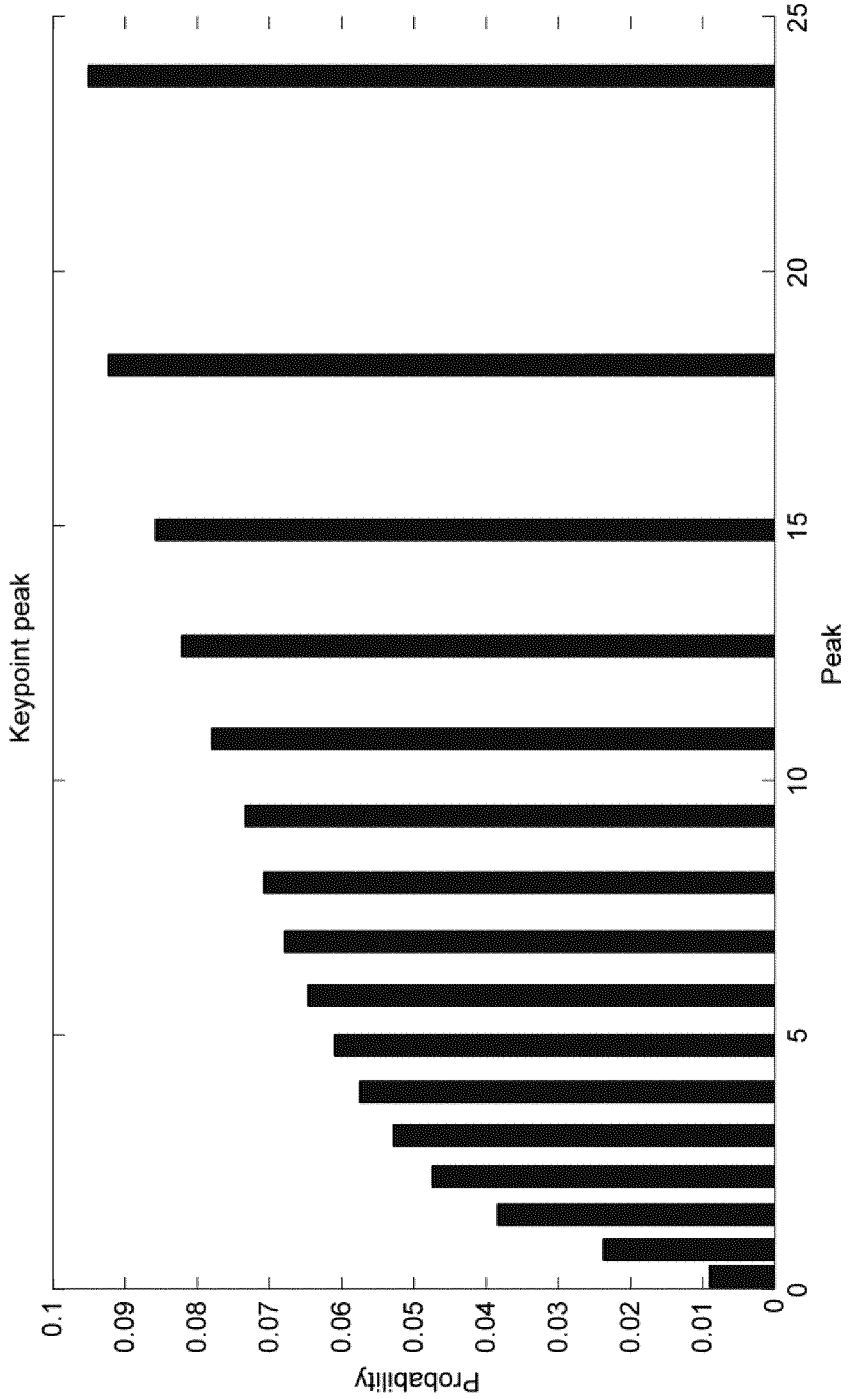

Keypoint peak
Peak
Probability
0
5
10
15
20
25
0.1
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01

FIG.2C

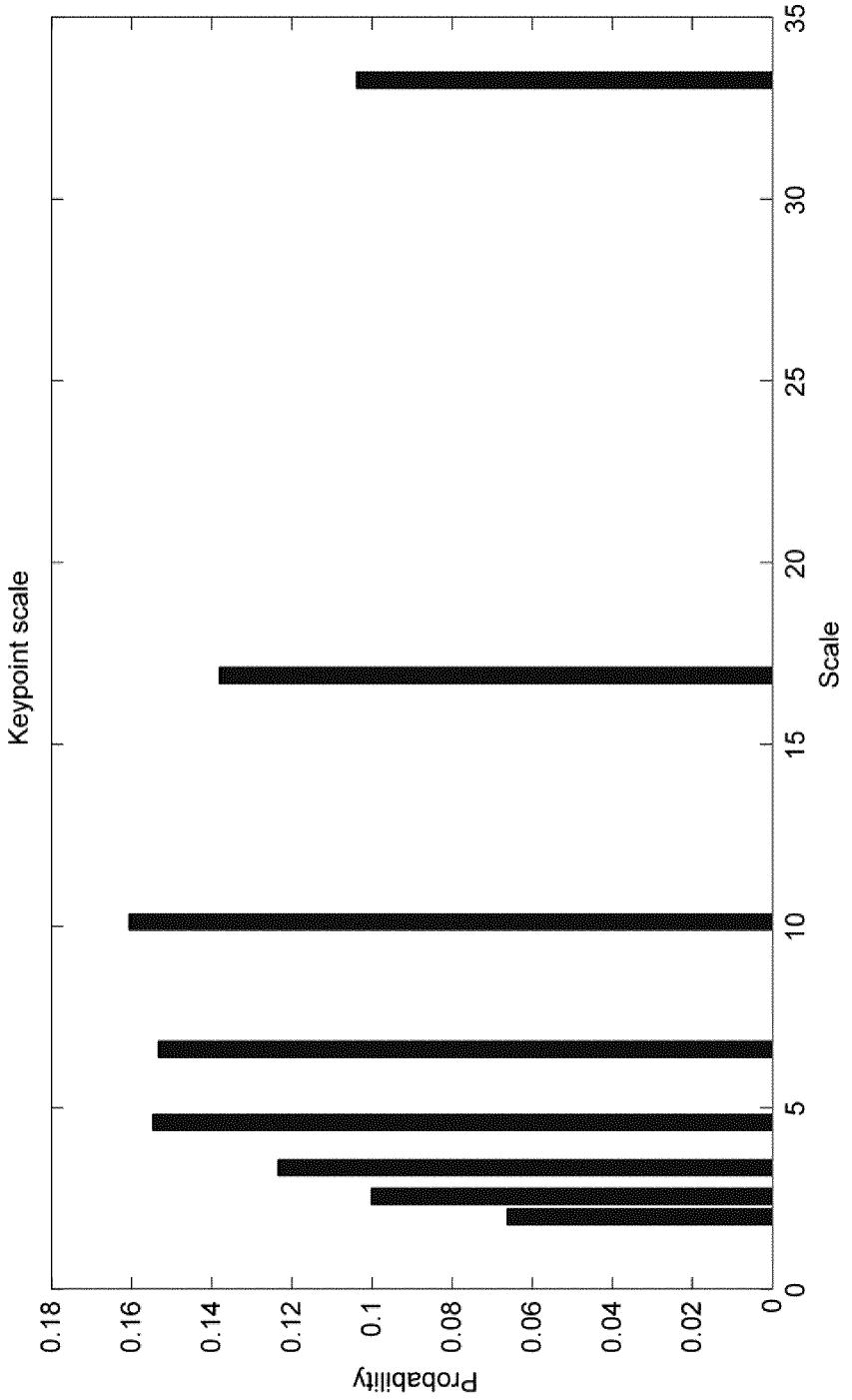

Keypoint scale
Scale
Probability
0
5
10
15
20
25
30
35
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18

FIG.2D

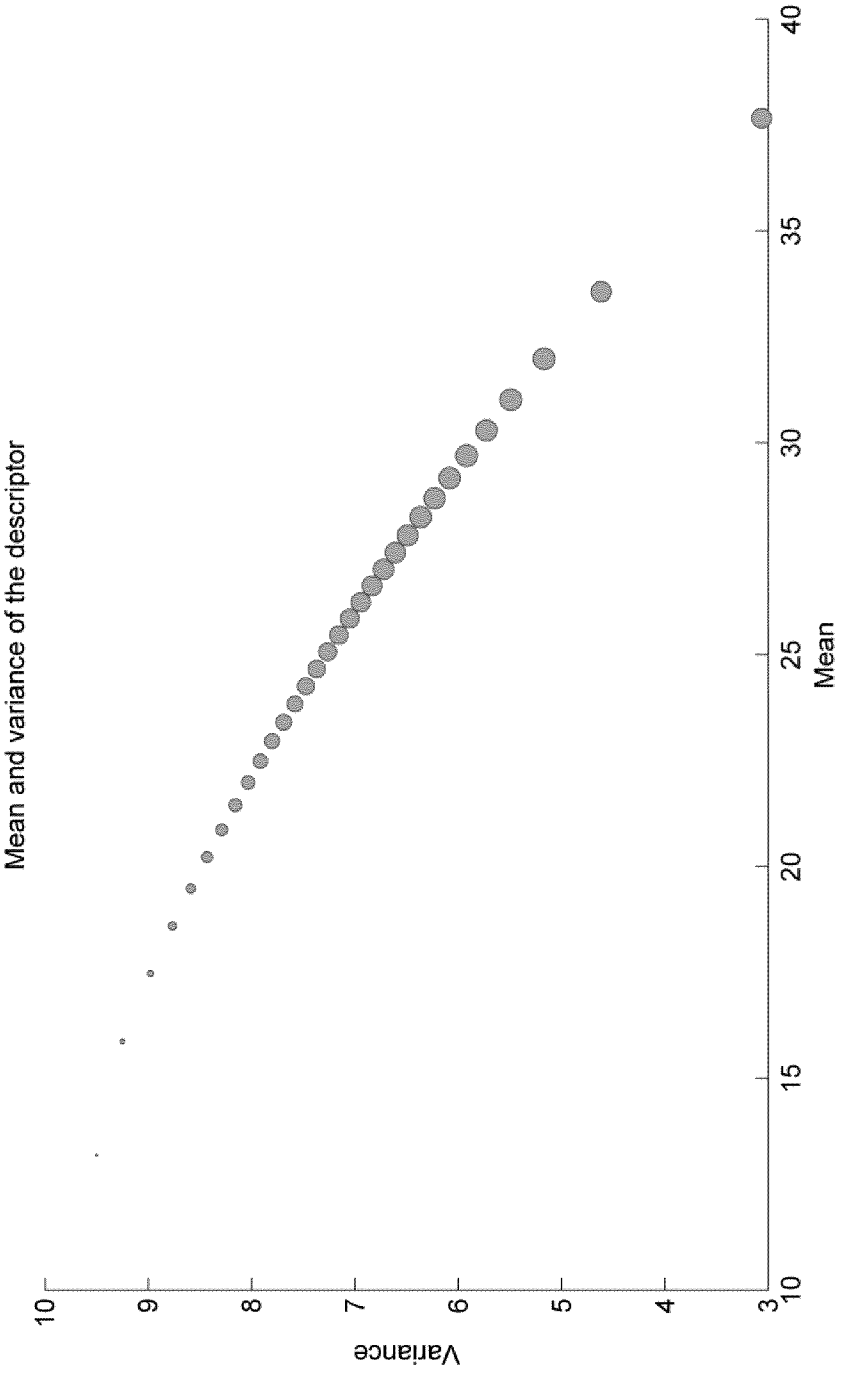

Mean and variance of the descriptor
Variance
Mean
10
9
8
7
6
5
4
3
10
15
20
25
30
35
40

FIG.2E

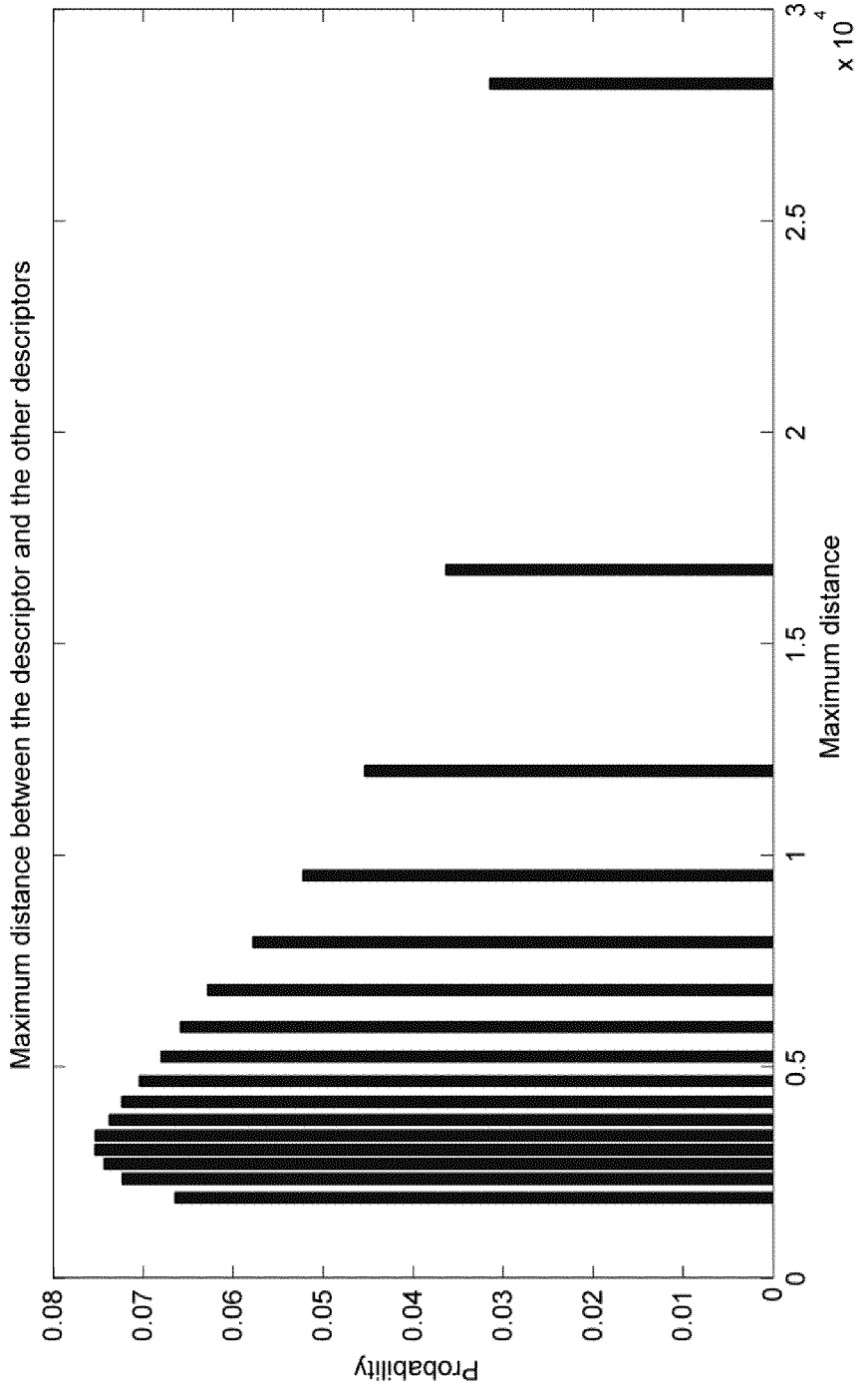

Maximum distance between the descriptor and the other descriptors
Probability
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
0.5
1
1.5
2
2.5
3
x 10^4
Maximum distance

FIG.2F

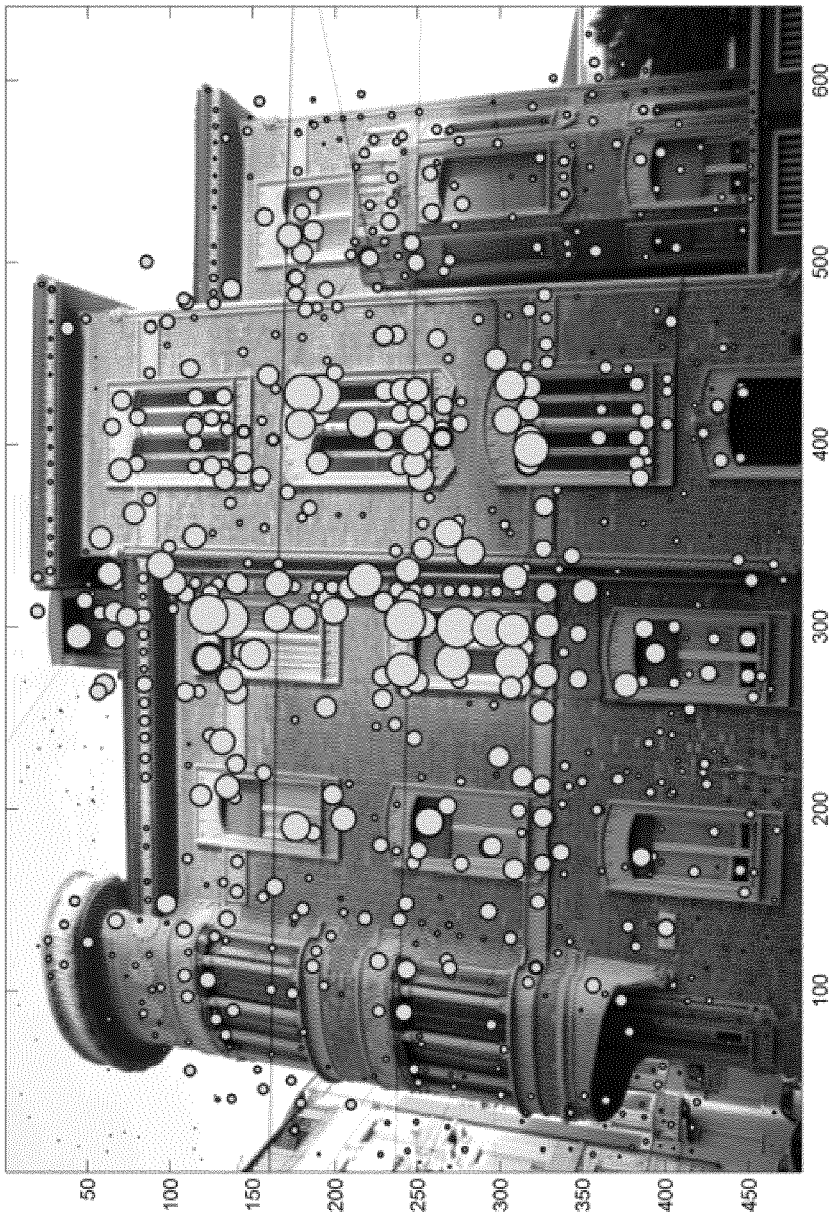


FIG.2G

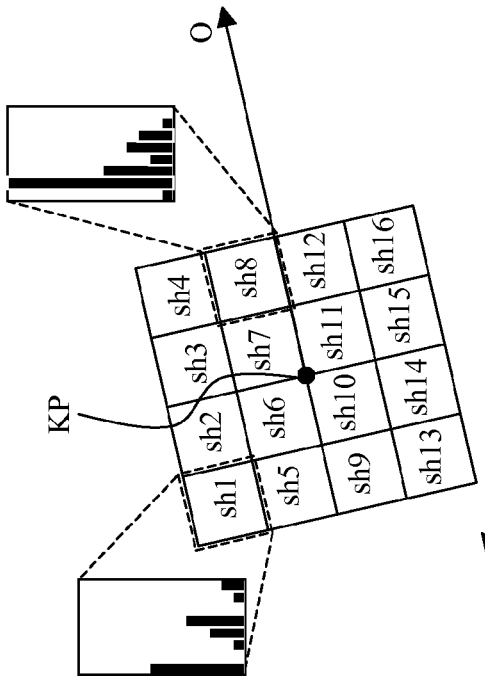

O
KP
sh1
sh2
sh3
sh4
sh5
sh6
sh7
sh8
sh9
sh10
sh11
sh12
sh13
sh14
sh15
sh16
D

FIG.3A

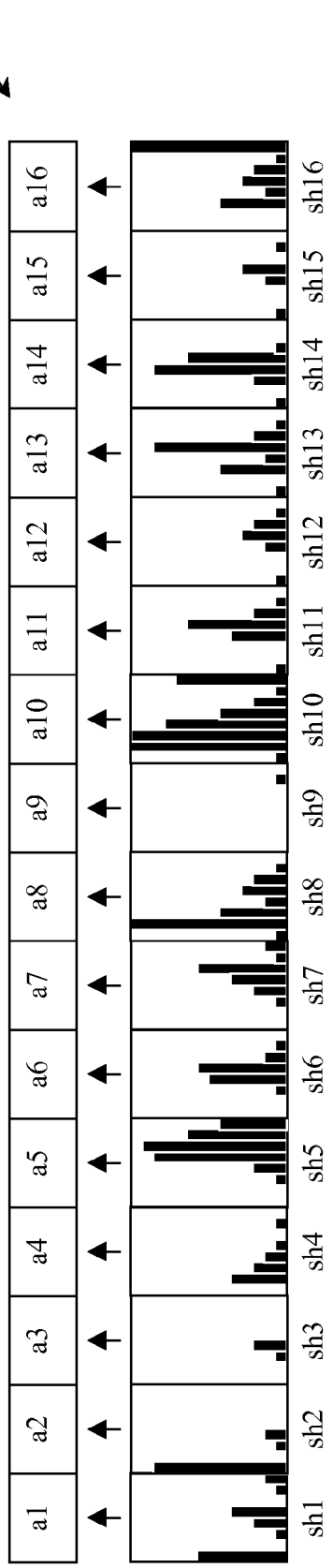

DA
a1
a2
a3
a4
a5
a6
a7
a8
a9
a10
a11
a12
a13
a14
a15
a16
sh1
sh2
sh3
sh4
sh5
sh6
sh7
sh8
sh9
sh10
sh11
sh12
sh13
sh14
sh15
sh16

FIG.3B

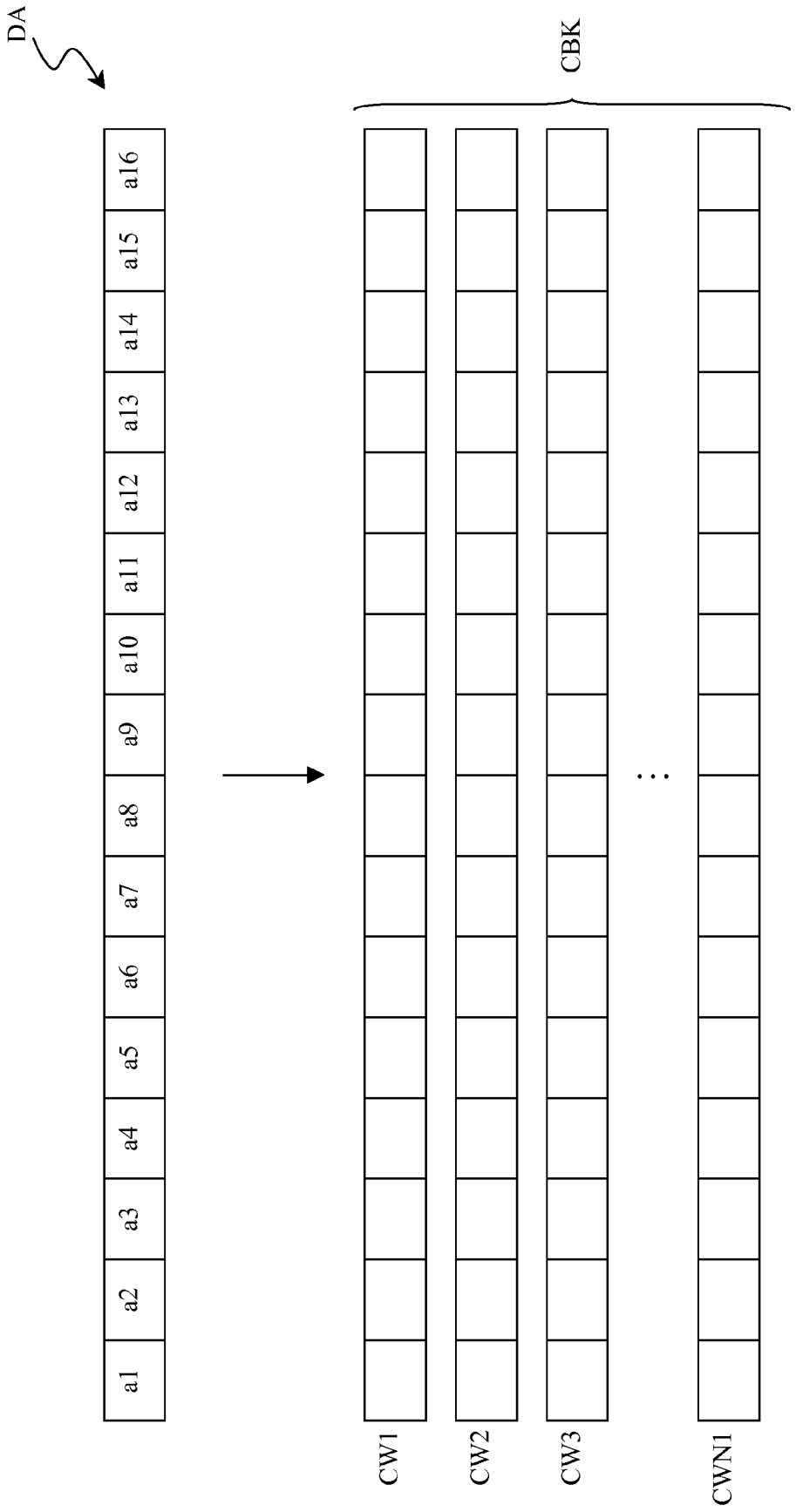

DA
CBK
a1
a2
a3
a4
a5
a6
a7
a8
a9
a10
a11
a12
a13
a14
a15
a16
CW1
CW2
CW3
CWN1

FIG.4A

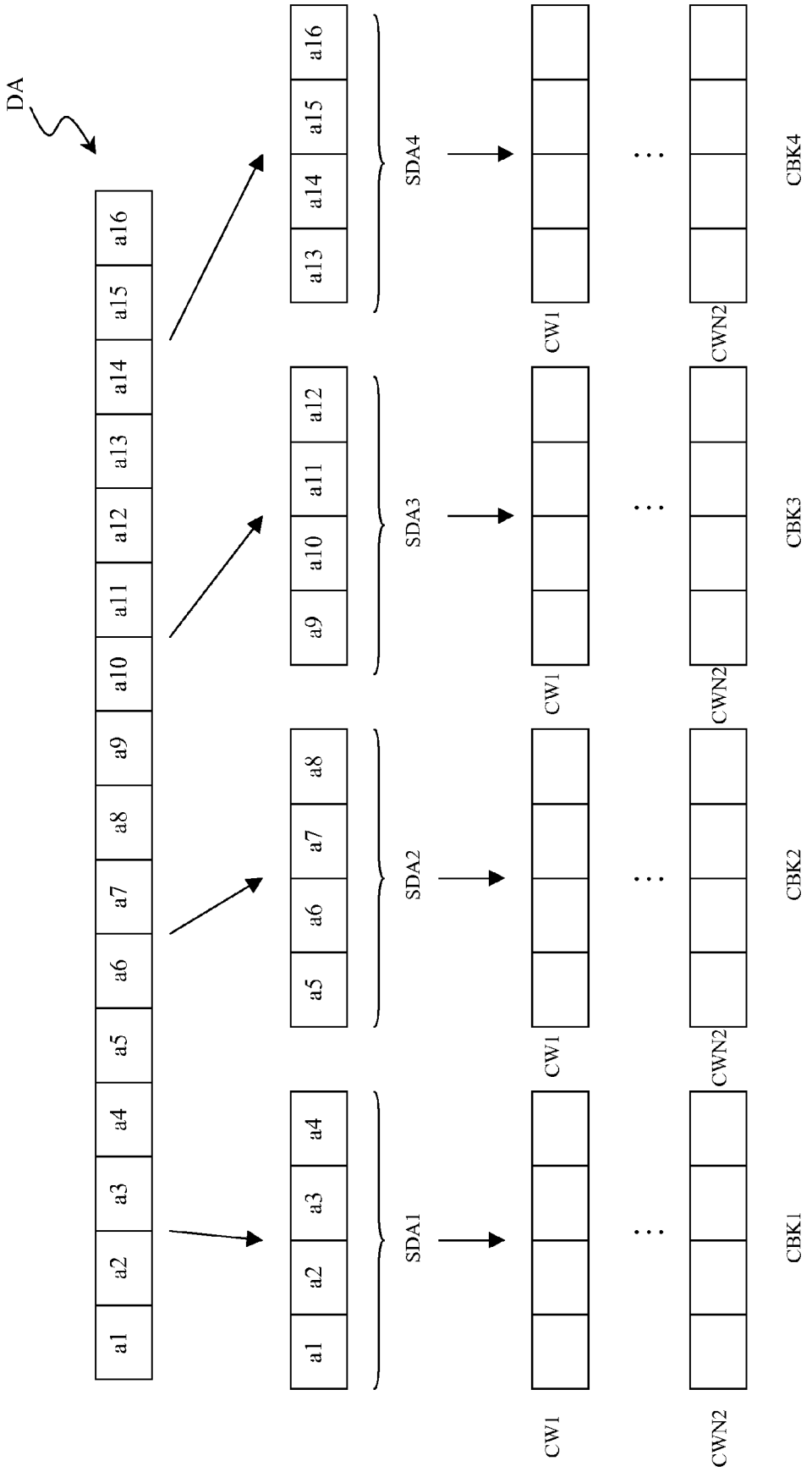

DA
a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16
SDA1
SDA2
SDA3
SDA4
CW1
CWN2
CBK1
CBK2
CBK3
CBK4

FIG.4B

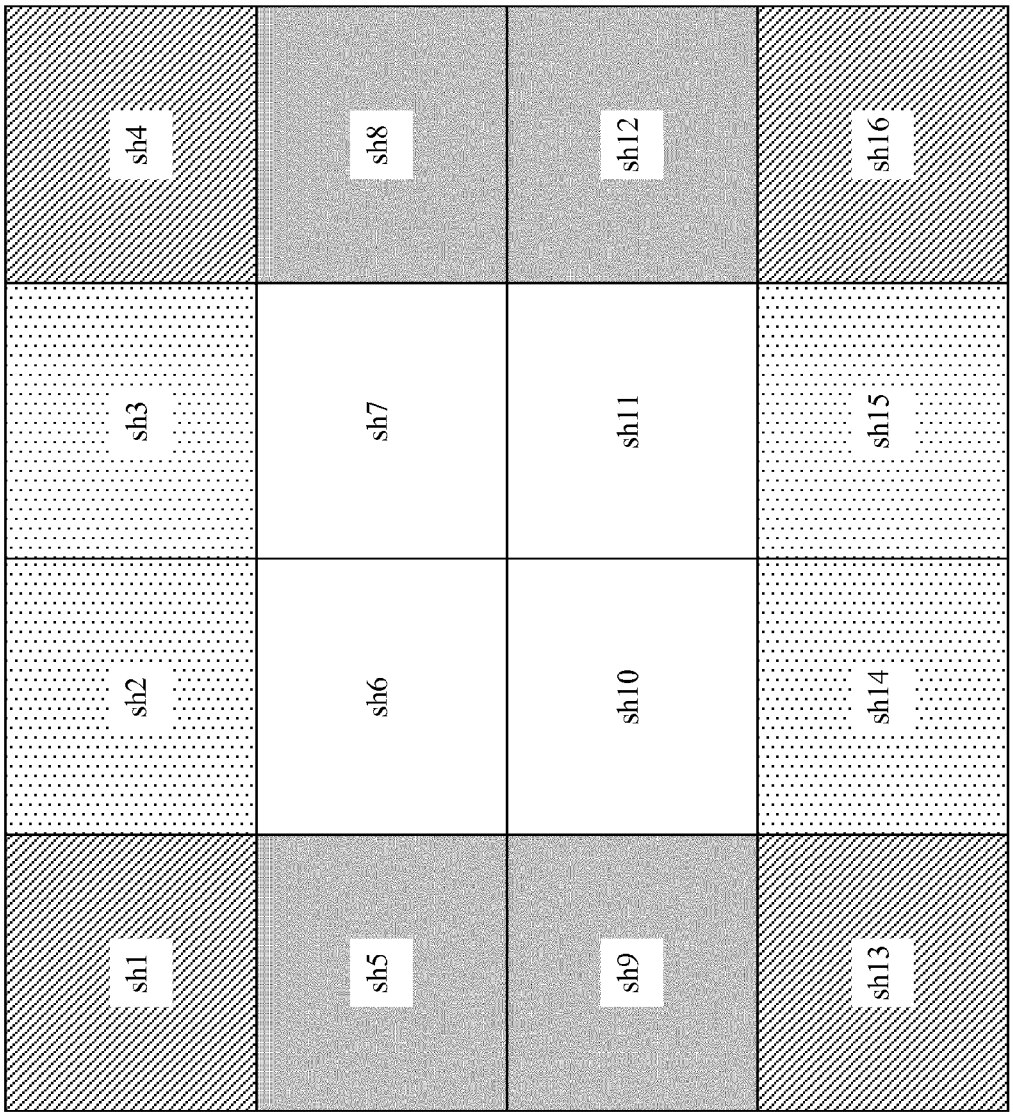

CF1: sh1, sh4, sh13, sh16
CF2: sh2, sh3, sh14, sh15
CF3: sh5, sh8, sh9, sh12
CF4: sh6, sh7, sh10, sh11
sh1
sh4
sh13
sh16
sh2
sh3
sh14
sh15
sh5
sh8
sh9
sh12
sh6
sh7
sh10
sh11

FIG.5

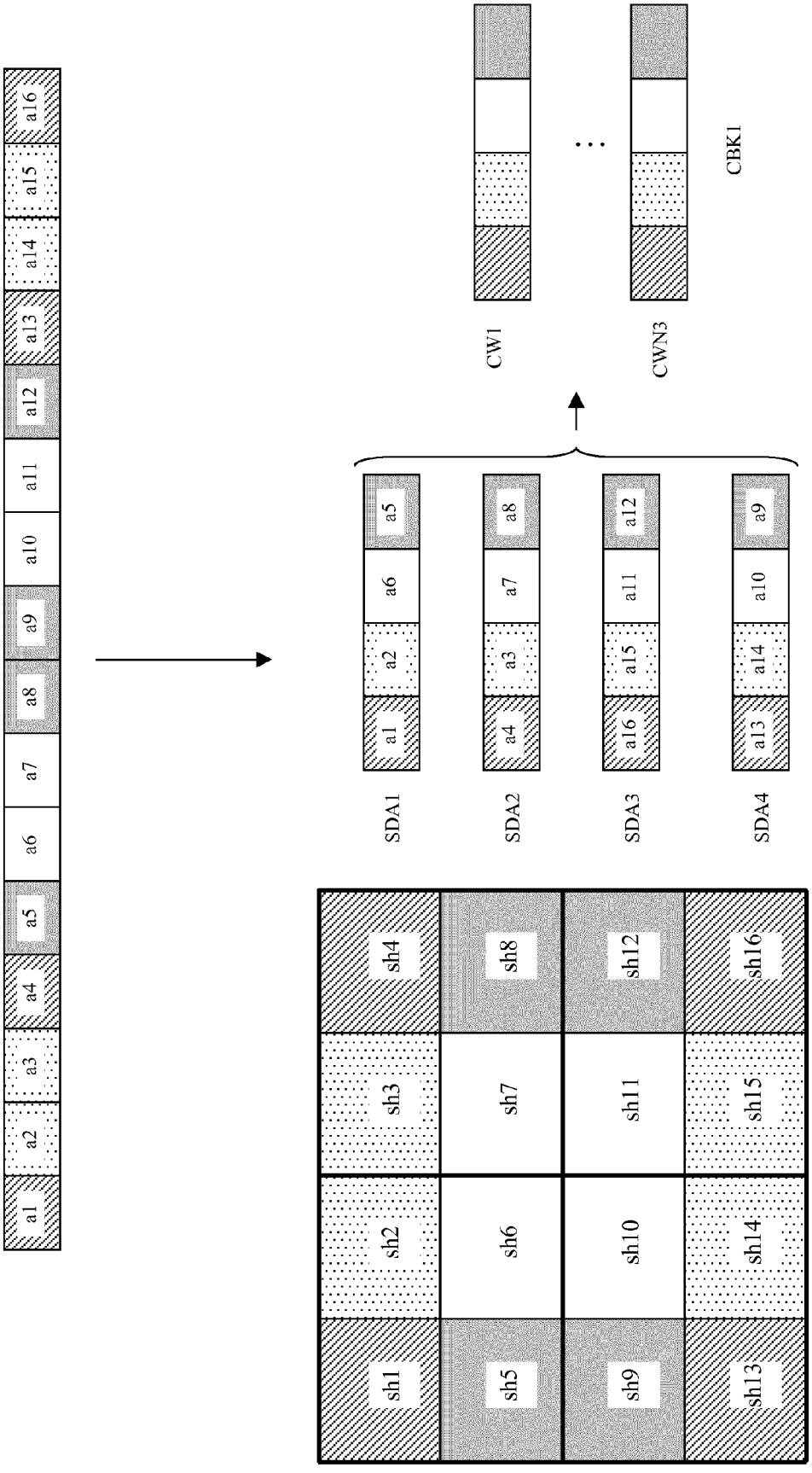

a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16
SDA1 a1 a2 a6 a5
SDA2 a4 a3 a7 a8
SDA3 a16 a15 a11 a12
SDA4 a13 a14 a10 a9
CW1 CWN3 CBK1
sh1 sh2 sh3 sh4
sh5 sh6 sh7 sh8
sh9 sh10 sh11 sh12
sh13 sh14 sh15 sh16

FIG.6A

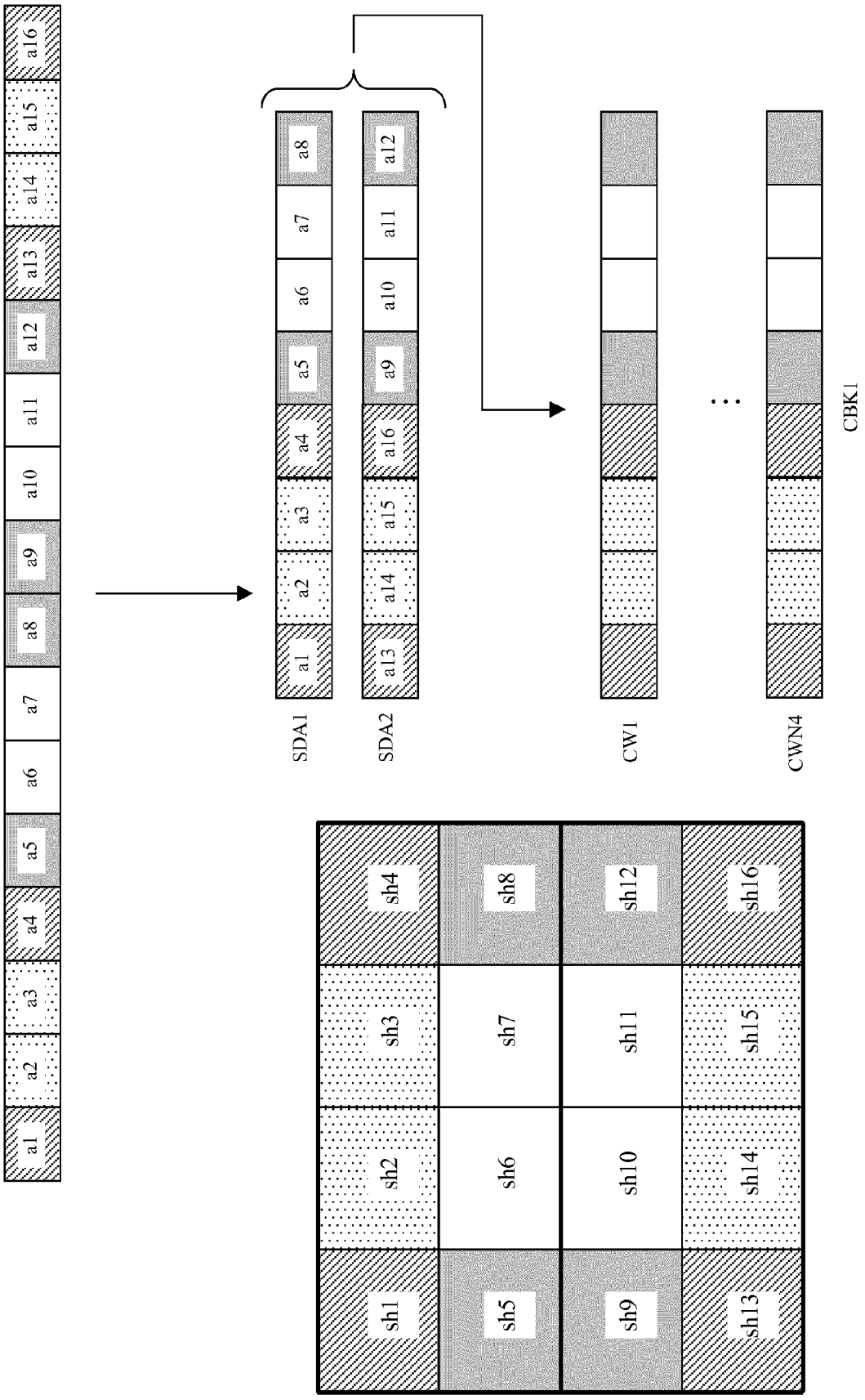

a1
a2
a3
a4
a5
a6
a7
a8
a9
a10
a11
a12
a13
a14
a15
a16
SDA1
SDA2
CW1
CWN4
CBK1
sh1
sh2
sh3
sh4
sh5
sh6
sh7
sh8
sh9
sh10
sh11
sh12
sh13
sh14
sh15
sh16

FIG.6B

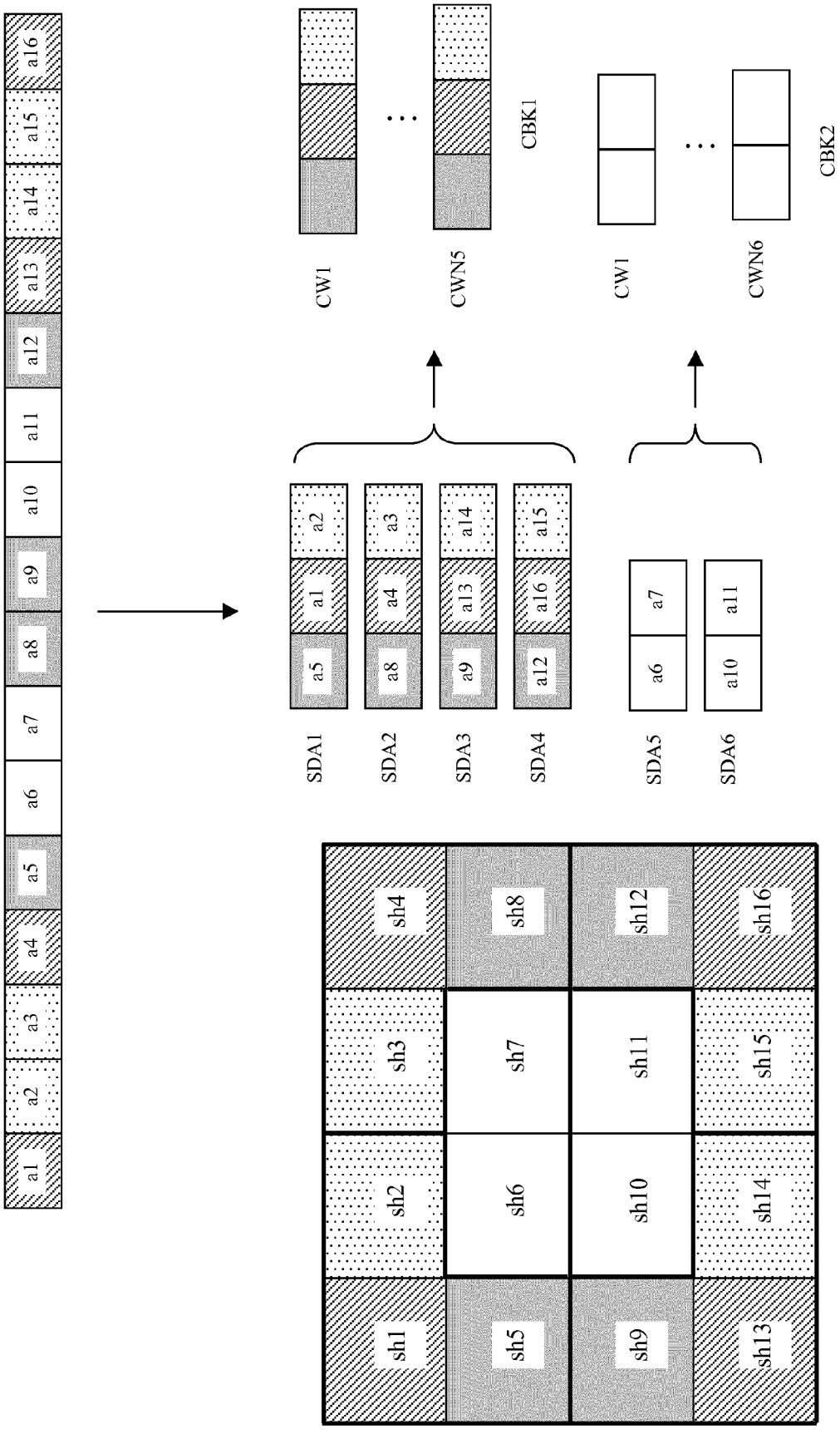

a1
a2
a3
a4
a5
a6
a7
a8
a9
a10
a11
a12
a13
a14
a15
a16
SDA1
SDA2
SDA3
SDA4
SDA5
SDA6
CW1
CWN5
CW1
CWN6
CBK1
CBK2
sh1
sh2
sh3
sh4
sh5
sh6
sh7
sh8
sh9
sh10
sh11
sh12
sh13
sh14
sh15
sh16

FIG.6C

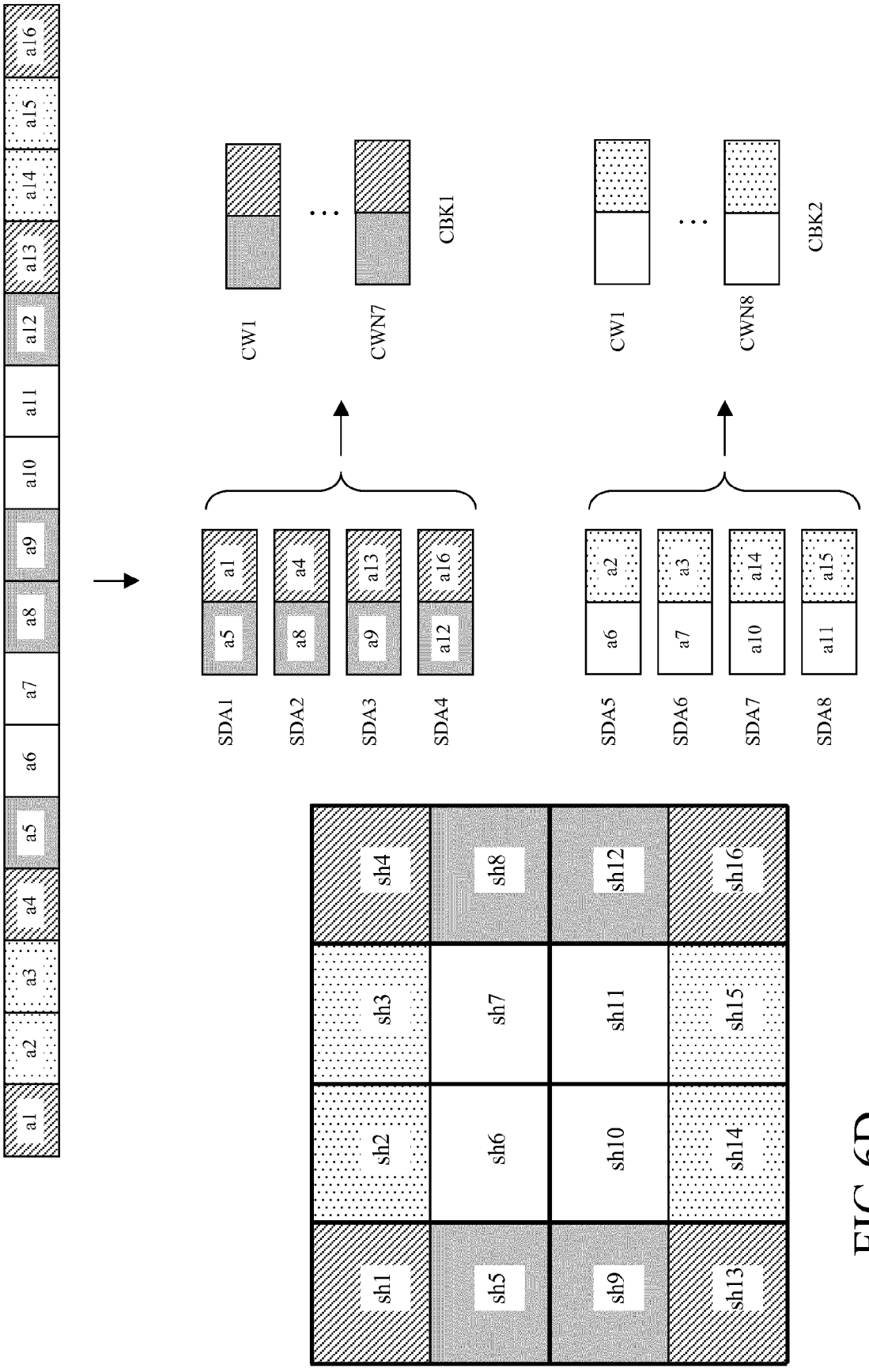

a1
a2
a3
a4
a5
a6
a7
a8
a9
a10
a11
a12
a13
a14
a15
a16
CW1
CWN7
CBK1
CW1
CWN8
CBK2
SDA1
SDA2
SDA3
SDA4
SDA5
SDA6
SDA7
SDA8
sh1
sh2
sh3
sh4
sh5
sh6
sh7
sh8
sh9
sh10
sh11
sh12
sh13
sh14
sh15
sh16

FIG.6D

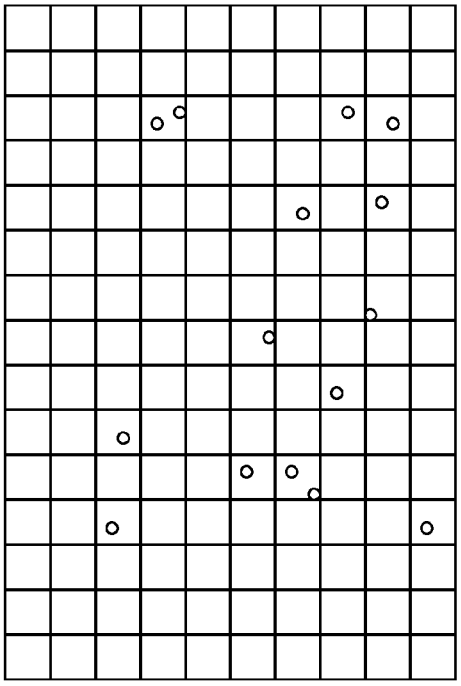


FIG.7B

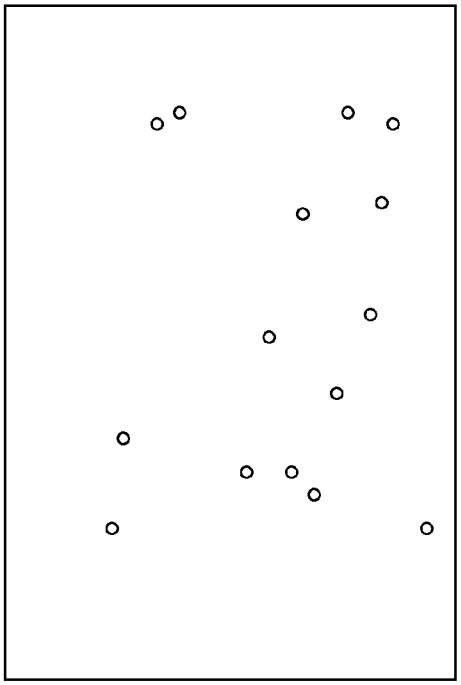


FIG.7A

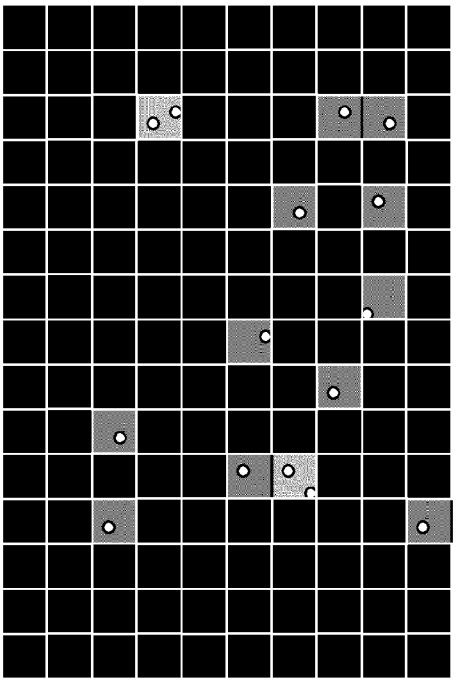


FIG.7C

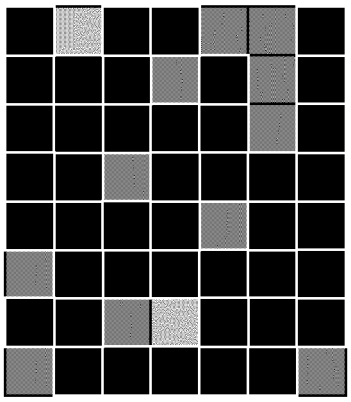


FIG. 7E

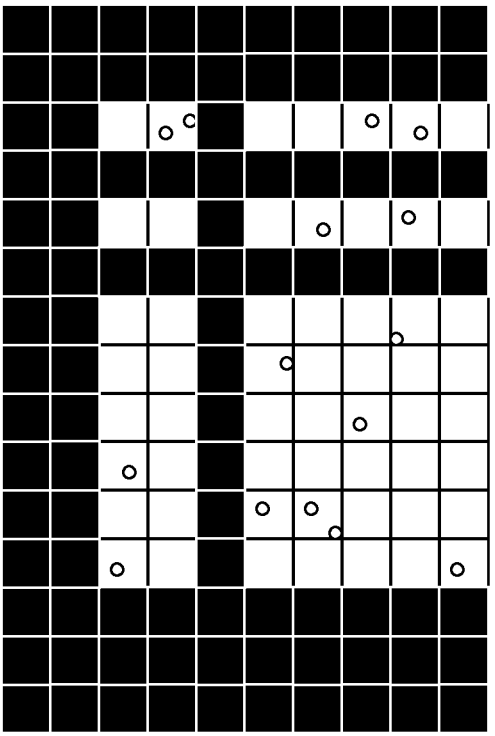


FIG. 7D

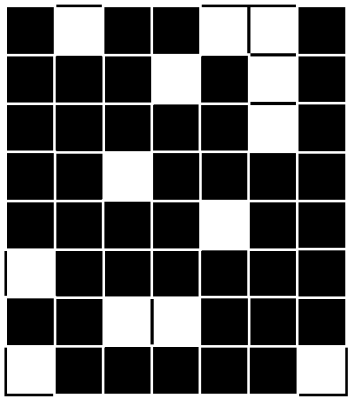


FIG. 7F

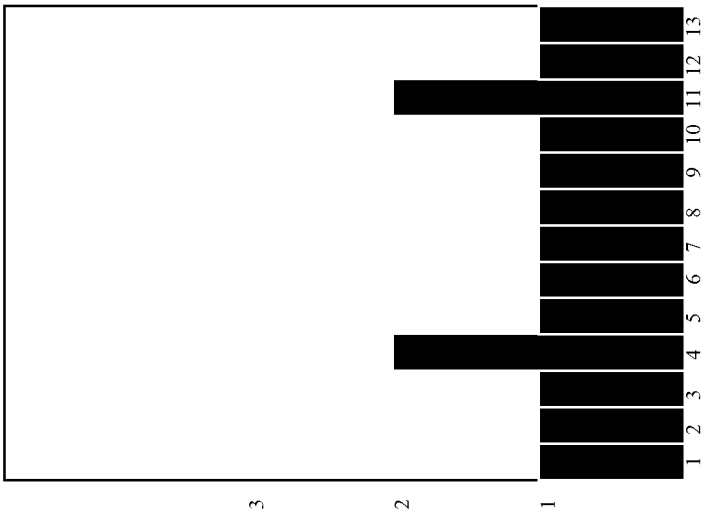


FIG.8B

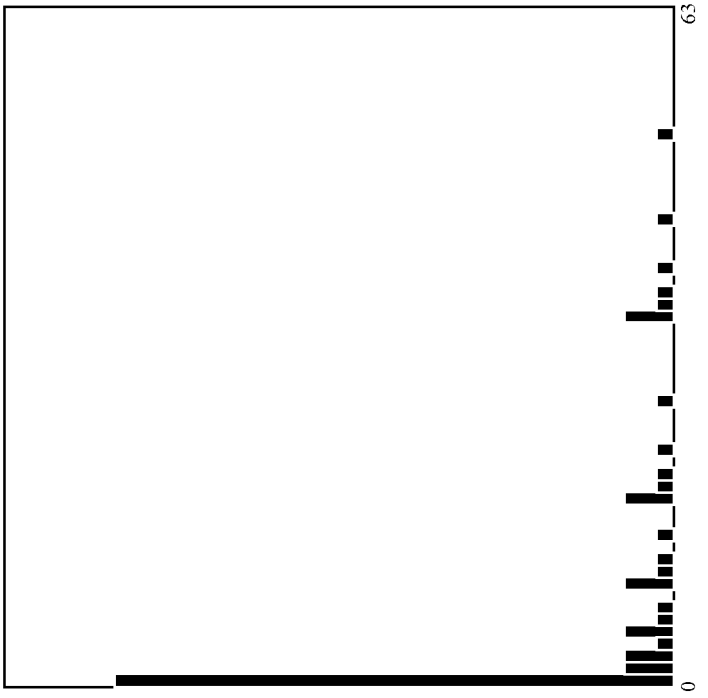


FIG.8A

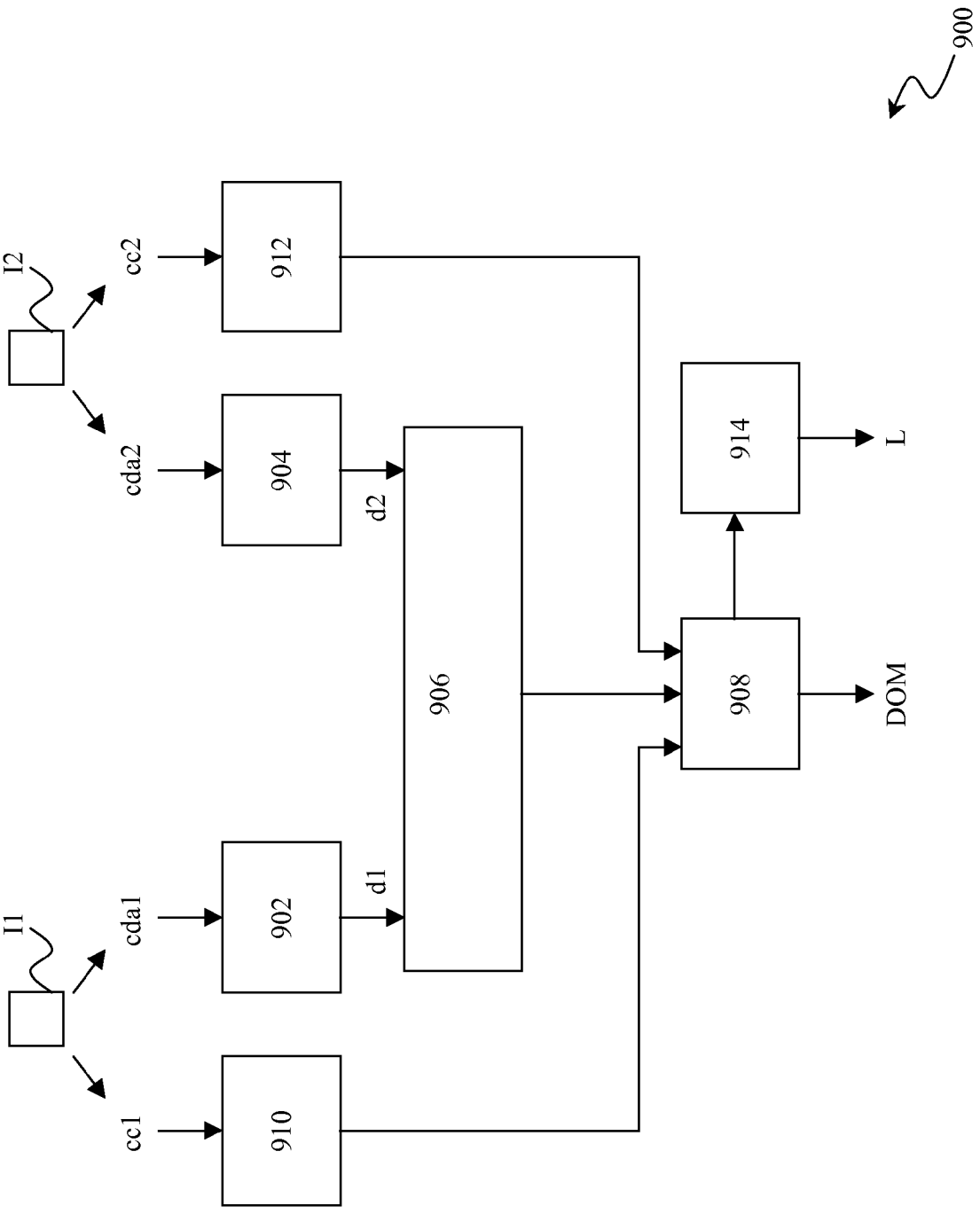

900
I2
cc2
912
cda2
904
d2
906
914
L
908
DOM
I1
cda1
902
d1
cc1
910

FIG.9

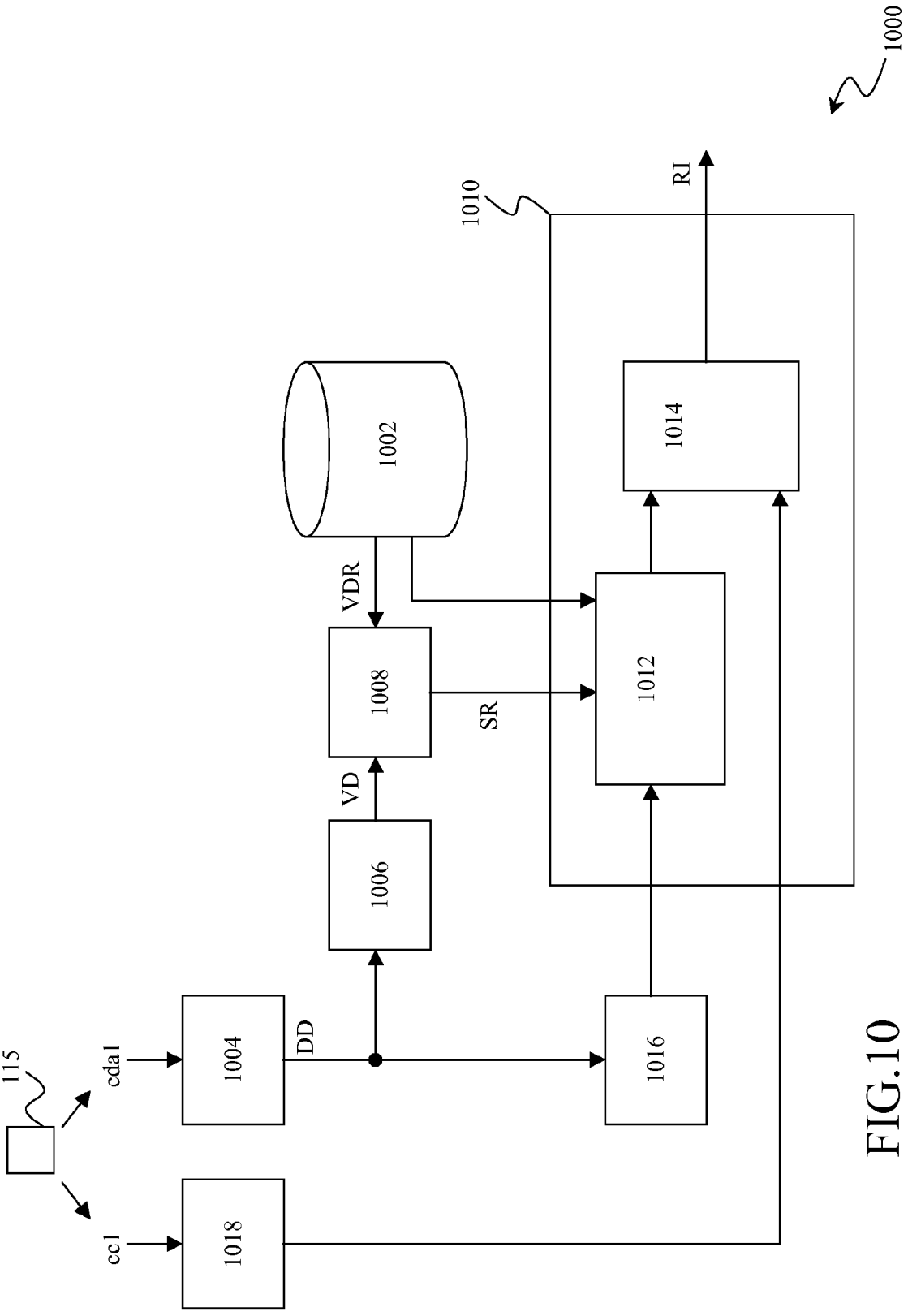

1000
1010
RI
1002
1014
VDR
1008
SR
1012
VD
1006
DD
1004
cda1
115
1016
cc1
1018

FIG.10

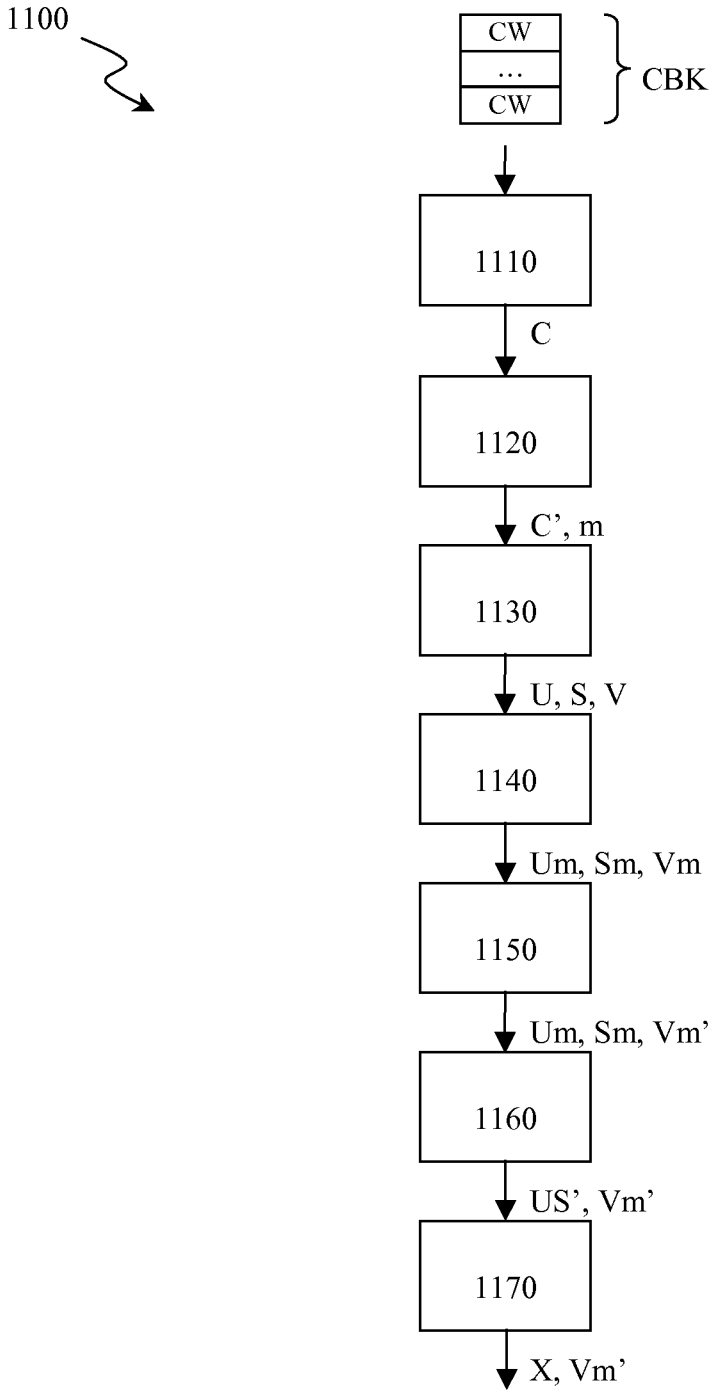

1100
CW
...
CW
CBK
1110
C
1120
C', m
1130
U, S, V
1140
Um, Sm, Vm
1150
Um, Sm, Vm'
1160
US', Vm'
1170
X, Vm'

FIG.11

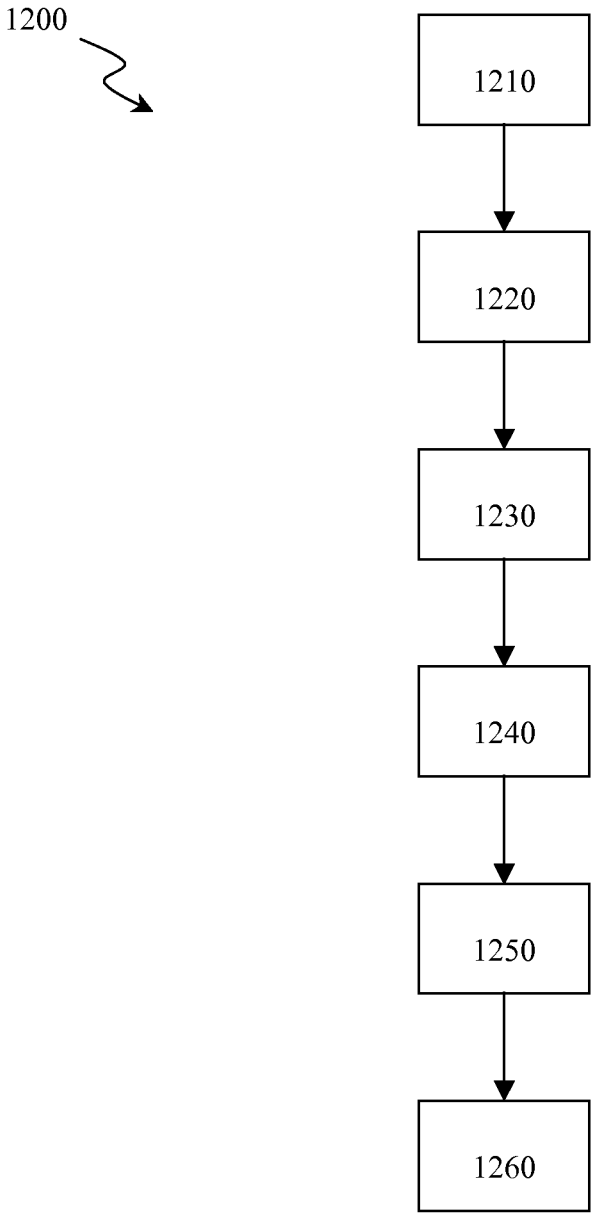


FIG.12

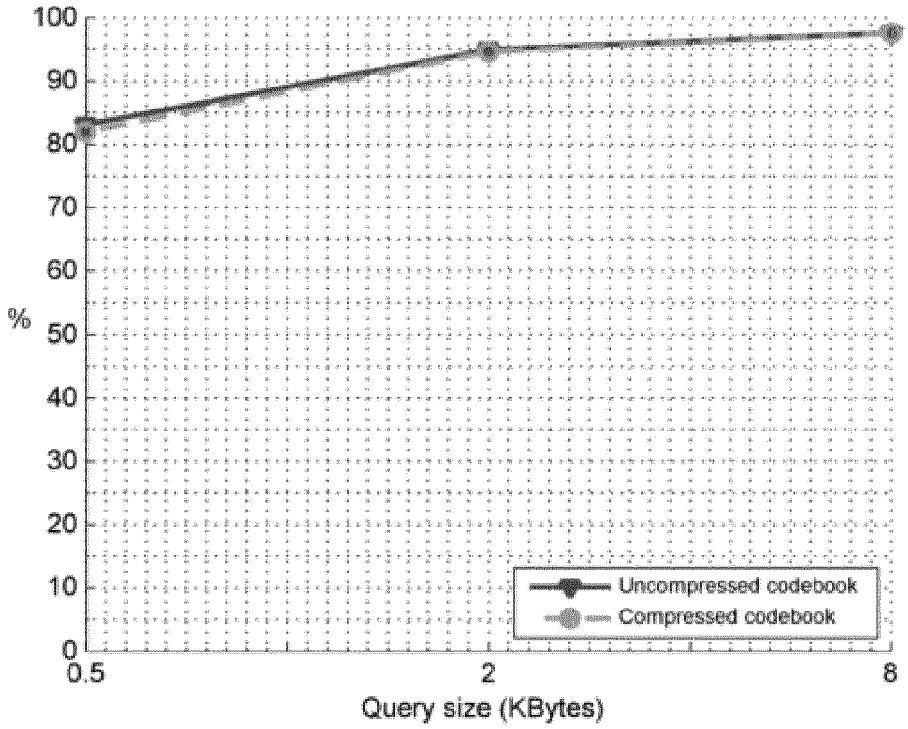


FIG.13A

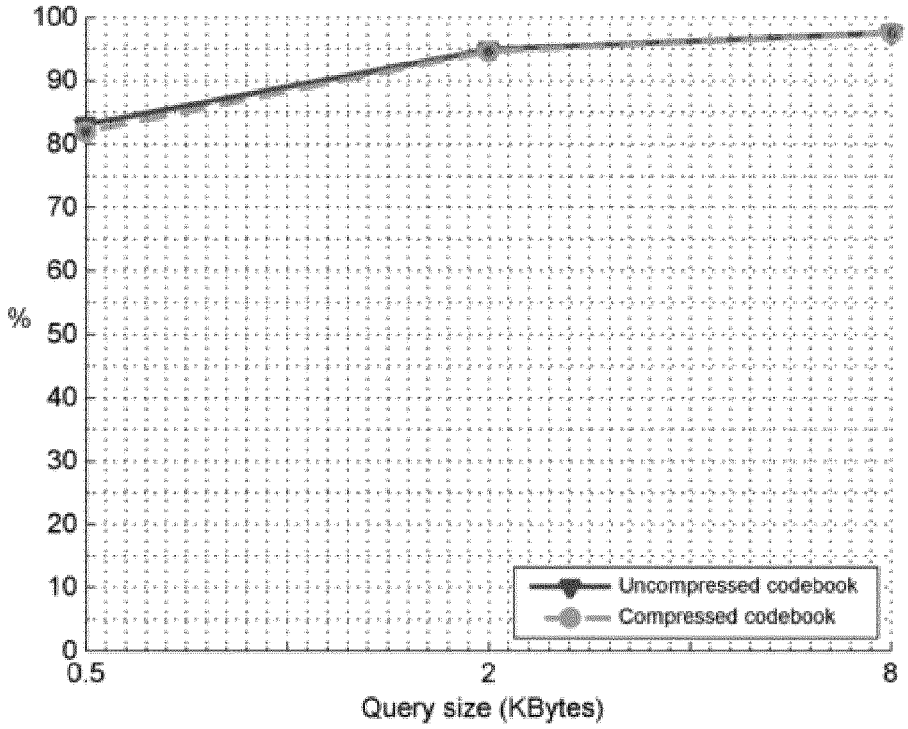


FIG.13B

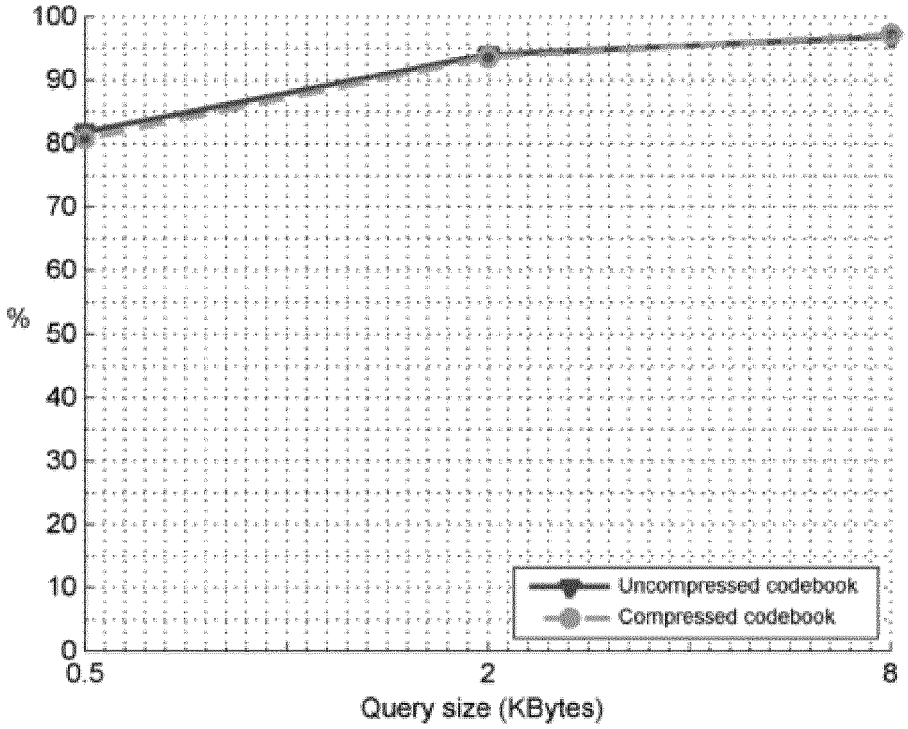


FIG.13C

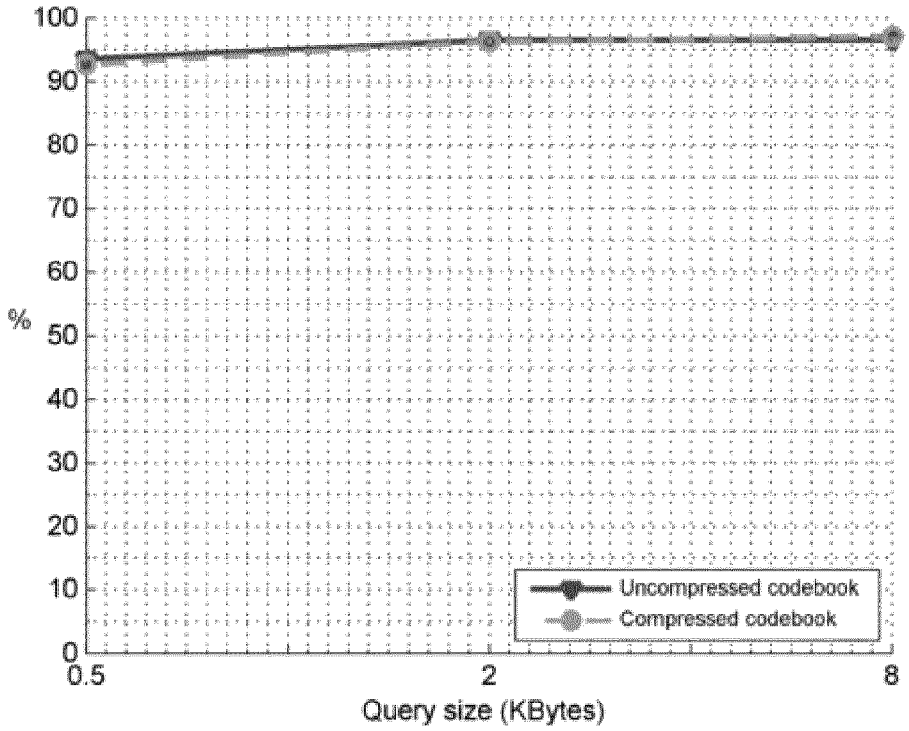


FIG.13D

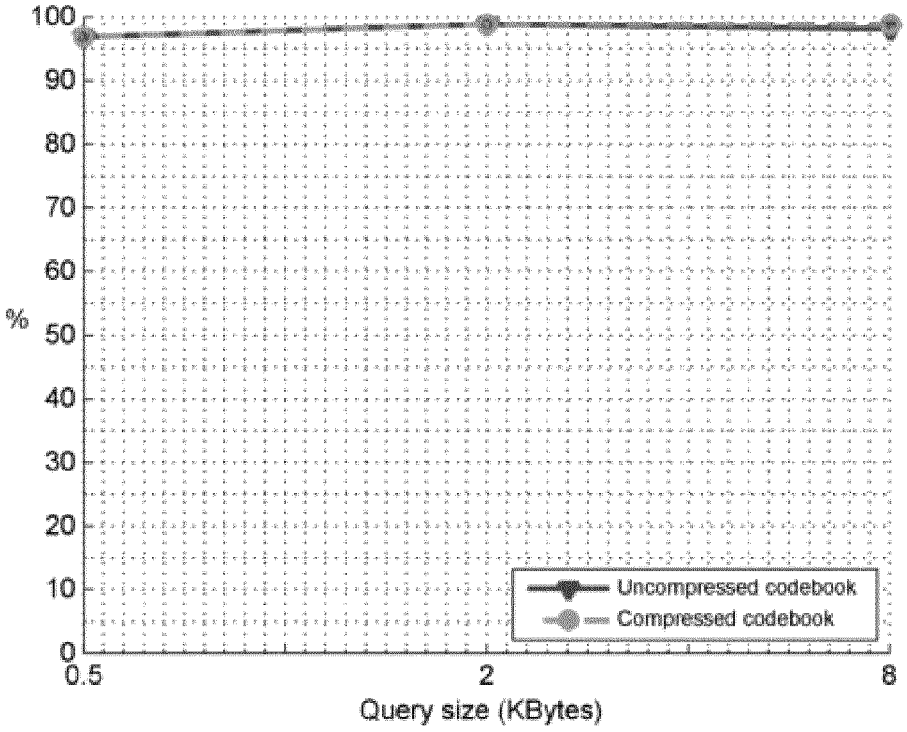


FIG.13E

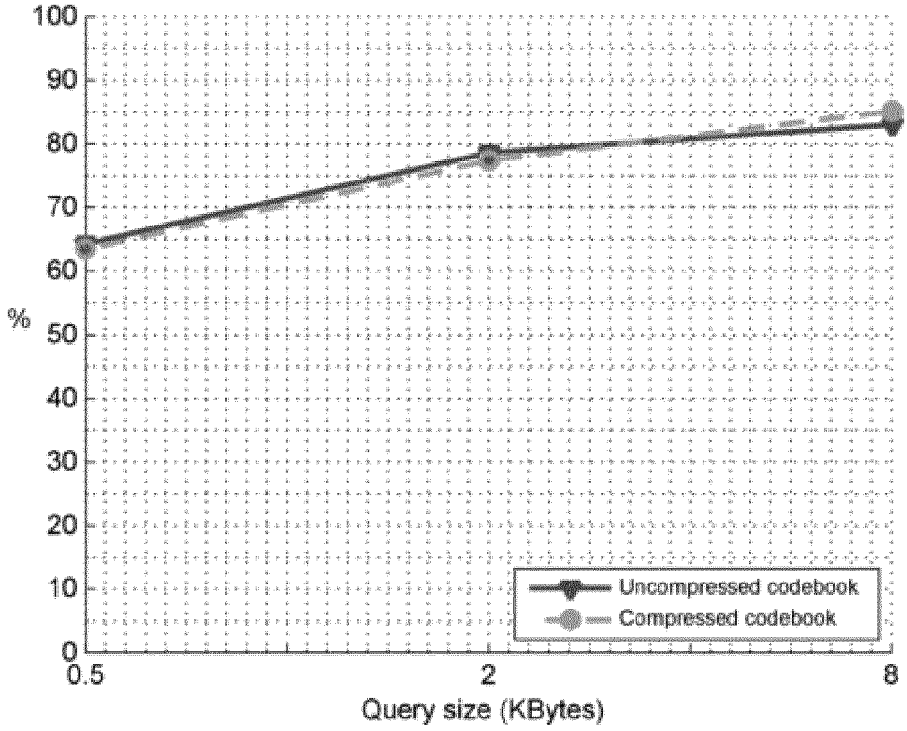


FIG.13F

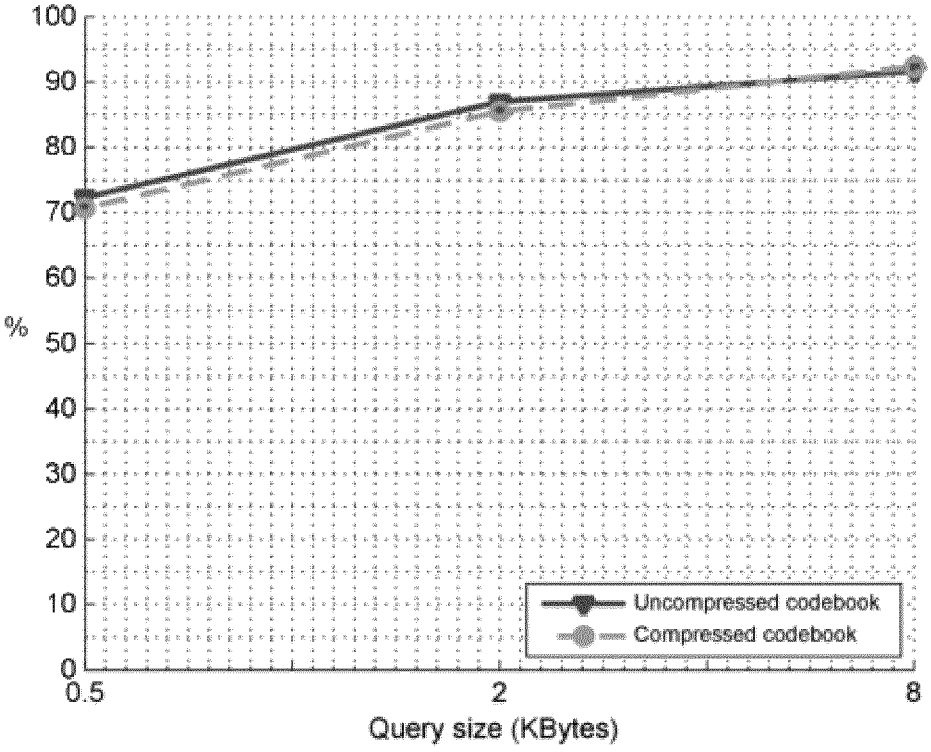

100
90
80
70
60
%
50
40
30
20
10
0
0.5
2
8
Query size (KBytes)
Uncompressed codebook
Compressed codebook

FIG.13G

METHOD AND SYSTEM FOR IMAGE ANALYSIS

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates to the field of the image analysis.

2. Description of the Related Art

In the field of the image analysis, a common operation provides for comparing two images in order to find the relation occurring therebetween in case both the images include at least a portion of a same scene or of a same object.

Among a high number of applications, the image comparison is of the utmost importance for calibrating video cameras belonging to a multi-camera system, for assessing the motion occurring between two frames of a video shoot, and for the recognition of an object within an image (e.g., a picture). The latter application is now assuming more and more importance due to the recent development of object recognition algorithms specifically designed to be employed in the so-called visual searching engines, i.e., automated services that, starting from a picture, are capable of identifying the object(s) pictured therein and offering information related to the identified object(s). Examples of known services of this type include Google Goggles, Nokia Point&Find, and kooaba Smart Visuals. An object recognition application typically provides for comparing a first image—in jargon, referred to as "query image"—depicting an object to be recognized with a plurality of model images, each one depicting a respective known object; this allows to perform a comparison among the object depicted in the query image and the objects depicted in the model images.

The model images are typically arranged in a proper model database. For example, in case the object recognition is exploited in an online shopping scenario, each model image corresponds to an item offered by an online store (e.g., the picture of a book cover, a DVD cover and/or a CD cover). The number of model images included in a database of such type is quite high; for example, a model database of an online shopping service may include several millions of different model images.

A very efficient way for performing comparing operations between two images provides for selecting a set of points—in jargon, referred to as keypoints—in the first image and then matching each keypoint of the set to a corresponding keypoint in the second image. The selection of which point of the first image has to become a keypoint is advantageously carried out by extracting local features of the area of the image surrounding the point itself, such as for example the point extraction scale, the privileged orientation of the area, and the so called "descriptor". In the field of the image analysis, a descriptor of a keypoint is a mathematic operator describing the luminance gradient of an area of the image (called patch) centered at the keypoint, with such patch that is orientated according to the main luminance gradient of the patch itself.

In "Distinctive image features from scale-invariant keypoints" by David G. Lowe, *International Journal of computer vision,* 2004, a Scale-Invariant Feature Transform (SIFT) descriptor has been proposed; briefly, in order to allow a reliable image recognition, the SIFT descriptors are generated taking into account that the local features extracted from the image corresponding to each keypoint should be detectable even under changes in image scale, noise and illumination. The SIFT descriptors are thus invariant to uniform scaling, orientation, and partially invariant to affine distortion and illumination changes.

The SIFT descriptor is a quite powerful tool, which allows to select keypoints for performing accurate image comparisons. However, this accuracy can be achieved only with the use of a quite large amount of data; for example, a typical SIFT descriptor is an array of 128 data bytes. Since the number of keypoints in each image is relatively high (for example, 1000-1500 keypoints for a standard VGA picture), and since each keypoint is associated with a corresponding SIFT descriptor, the overall amount of data to be processed may become excessive for being efficiently managed.

This drawback is exacerbated in case the scenario involves the use of mobile terminals (e.g., identification of objects extracted from pictures taken by the camera of a smarthpone). Indeed, since the operations to be performed for carrying out the image analysis are quite complex and demanding in terms of computational load, in this case most of the operations are usually performed at the server side; in order to have all the information required to perform the analysis, the server needs to receive from the mobile terminal all the required data, including the SIFT descriptors for all the keypoints. Thus, the amount of data to be transmitted from the terminal to the server may become excessive for guaranteeing a good efficiency of the service.

According to a solution known in the art, such as for example the one employed by Google Goggles, this drawback is solved at the root by directly transmitting the image, and not the descriptors, from the mobile terminal to the server. Indeed, because of the quite high number of keypoints, the amount of data of the corresponding SIFT descriptors may exceed the size (in terms of bytes) of a standard VGA picture itself.

SUMMARY OF THE INVENTION

The Applicant has found that the approaches known in the art are not efficient, still requiring the management of a high amount of data, limiting the scalability of the system and the overall performances.

The Applicant has tackled the problem of how to improve these approaches in terms of amount of data to be processed.

In particular, the Applicant has tackled the problem to provide a method for processing an image which requires a reduced amount of data to be managed.

The Applicant has found that the above problem is solved by compressing the codebook matrix through factorisation of the codebook matrix and truncation of the matrices resulting from said factorisation. An aspect of the present invention relates to a method for processing an image, comprising:

identifying a group of keypoints in the image;

for each keypoint of the group

a) calculating a corresponding descriptor array including a plurality of array elements, each array element storing values taken by a corresponding color gradient histogram of a respective sub-region of the image in the neighborhood of the keypoint;

b) generating at least one compressed descriptor array by compressing at least one portion of the descriptor array by means of vector quantization using a codebook comprising a plurality of codewords;

exploiting said at least one compressed descriptor array of the keypoints of said group for analysing the image, wherein the method further comprises:

compressing the codebook, said compressing the codebook including:

generating a codebook matrix, each row of the codebook matrix being a codeword of the codebook;

factorising the codebook matrix so as to obtain the product of at least a first matrix and a second matrix, the energy of the items of the second matrix generally non-increasing as the column indexes of such matrixes increase,

truncating the first matrix by removing therefrom a first number of last columns;

truncating the second matrix by removing therefrom the first number of last columns and the first number of last rows;

generating a first further matrix corresponding to the product of the truncated first matrix by the truncated second matrix;

quantizing each item of the first further matrix, wherein each item belonging to a column of the first further matrix is quantized using a corresponding number of quantization levels that is lower than or equal to the number of quantization levels used to quantize the items belonging to a preceding column:

generating a second further matrix wherein each item of the second further matrix corresponds to an item of the first further matrix, each item of said second further matrix being an index associated to the quantization level assumed by the corresponding quantized item of the first further matrix;

storing the codebook by memorizing said indexes of the second further matrix in a memory unit.

According to an embodiment of the present invention the first matrix (U) is an orthonormal matrix and the second matrix (S) is a diagonal matrix whose diagonal items correspond to the singular values of the codebook matrix (C), each diagonal item belonging to a column of the second matrix, except a last column, being higher then the item belonging to the following column.

According to an embodiment of the present invention said product of at least the first matrix (U) and the second matrix (S) further comprise the transpose of a third matrix (V), the third matrix (V) being an orthonormal matrix

and wherein said method further comprises:

truncating said third matrix by removing therefrom the first number of last columns;

storing the truncated third matrix in the memory unit.

According to an embodiment of the present invention said truncating the first, the second and the third matrices further comprises:

calculating the energy of the second matrix;

setting a first target value for the memory space occupied by all the quantized items of the second further matrix when memorized in the memory unit;

for each diagonal item of the second matrix, calculating a respective allocation value corresponding to the first target value multiplied by a ratio between the energy of such item and the energy of the second matrix;

rounding each allocation value to an integer value, and

setting the first number to the number of columns of the codebook matrix minus the higher column index of the second matrix for which the rounded allocation value corresponding to the diagonal item belonging to the column of the second matrix identified by such column index is higher than a threshold.

According to an embodiment of the present invention said quantizing each item of the first further matrix further comprises, for each column of the first further matrix, setting the corresponding number of quantization levels for quantizing each item of said column of the first further matrix to two raised to the power of the rounded allocation value corresponding to the diagonal item belonging to the column of the second matrix having the same column index as said column of the first further matrix.

According to an embodiment of the present invention the method further includes:

selecting a data type;

scaling each item of the truncated third matrix by a coefficient corresponding to the ratio between the highest value that can be represented with the selected data type and the highest absolute value among the absolute values of the items of the truncated third matrix, and

representing each scaled item of the truncated third matrix with the selected data type, said storing the codebook further including memorizing the scaled items of the tuncated third matrix represented with the selected data type in the memory unit.

According to an embodiment of the present invention said generating the first further matrix comprises multiplying the truncated first matrix by the truncated second matrix and then multiplying each item of the resulting matrix by the coefficient.

According to an embodiment of the present invention said factorizing the codebook matrix comprises using the Singular Value Decomposition.

According to an embodiment of the present invention said method further includes, before factorising the codebook matrix into the product of at least the first and second matrices, calculating an average array obtained from the average of all the rows of the codebook matrix and then subtracting such average array from each row of the codebook matrix, said storing the codebook further including memorizing the average array in the memory unit.

Another aspect of the present invention provides for a system for processing an image comprising a group of keypoints. The system being configured to perform the following operations:

for each keypoint of the group:

a) calculating a corresponding descriptor array including a plurality of array elements, each array element storing values taken by a corresponding color gradient histogram of a respective sub-region of the image in the neighborhood of the keypoint;

b) generating at least one compressed descriptor array by compressing at least one portion of the descriptor array by means of vector quantization using a codebook comprising a plurality of codewords;

exploiting said at least one compressed descriptor array of the keypoints of said group for analysing the image,

wherein the method further comprises:

compressing the codebook, said compressing the codebook including:

generating a codebook matrix, each row of the codebook matrix being a codeword of the codebook;

factorising the codebook matrix so as to obtain the product of at least a first matrix and a second matrix, the energy of the items of the second matrix generally non-increasing as the column indexes of such matrixes increase,

truncating the first matrix by removing therefrom a first number of last columns;

truncating the second matrix by removing therefrom the first number of last columns and the first number of last rows;

generating a first further matrix corresponding to the product of the truncated first matrix by the truncated second matrix;

quantizing each item of the first further matrix, wherein each item belonging to a column of the first further matrix is quantized using a corresponding number of quantization levels that is lower than or equal to the number of quantization levels used to quantize the items belonging to a preceding column:

generating a second further matrix wherein each item of the second further matrix corresponds to an item of the first further matrix, each item of said second further matrix being an index associated to the quantization level assumed by the corresponding quantized item of the first further matrix;

storing the codebook by memorizing said indexes of the second further matrix in a memory unit.

Another aspect of the present invention provides for a software program product including code portions configured to perform the method according to the first aspect or to any of the embodiments.

BRIEF DESCRIPTION OF THE DRAWINGS

These and other features and advantages of the present invention will be made evident by the following description of some exemplary and non-limitative embodiments thereof, to be read in conjunction with the attached drawings, wherein:

FIG. **1** illustrates in terms of functional blocks an extraction procedure directed to extract from a query image an optimal set of keypoints and generate a compressed set of descriptors according to an embodiment of the present invention;

FIGS. **2**A-**2**F are statistical distributions of corresponding selected local features of keypoints according to some exemplary embodiments of the present invention;

FIG. **2**G is an exemplary picture processed according to the extraction procedure of FIG. **1**;

FIG. **3**A illustrates an exemplary descriptor of the SIFT type;

FIG. **3**B illustrates an exemplary descriptor array of the descriptor of FIG. **3**A;

FIG. **4**A illustrates an exemplary descriptor array compression according to a solution known in the art;

FIG. **4**B illustrates an exemplary descriptor array compression according to another solution known in the art;

FIG. **5** illustrates an arrangement of sub-histograms of a descriptor in correlation families according to an embodiment of the present invention;

FIGS. **6**A-**6**D show how the descriptor array is compressed according to exemplary embodiments of the present invention;

FIG. **7**A illustrates an exemplary distribution of keypoints KP;

FIG. **7**B illustrates how a grid can be superimposed over the query image for quantizing the coordinates of the keypoints of FIG. **7**A;

FIG. **7**C is an exemplary graphical depiction of a histogram obtained by superimposing the grid of FIG. **7**B over the set of keypoints KP of FIG. **7**A;

FIG. **7**D identifies the columns and rows of the grid of FIG. **7**B which are entirely formed by cells that do not include any keypoint;

FIG. **7**E illustrates an exemplary histogram over a rank-1 support;

FIG. **7**F illustrates a histogram map corresponding to the histogram over the rank-1 support of FIG. **7**E;

FIG. **8**A illustrates an example of a word histogram;

FIG. **8**B illustrates an example of a histogram map;

FIG. **9** illustrates in terms of functional blocks a matching procedure directed to perform the comparison between two images according to an embodiment of the present invention;

FIG. **10** illustrates in terms of functional blocks a retrieval procedure directed to retrieve from a model database a model image depicting the same object/scene depicted in the query image according to an embodiment of the present invention;

FIG. **11** is a flow chart illustrating the main phases of a codebook compression procedure according to an embodiment of the present invention;

FIG. **12** is a flow chart illustrating the main phases of a method for selecting some parameter values to be used in the codebook compression procedure according to an embodiment of the present invention;

FIGS. **13**A-**13**G illustrates performance of image comparison systems based on non compressed codebooks and based on compressed codebooks.

DETAILED DESCRIPTION OF EXEMPLARY EMBODIMENTS OF THE INVENTION

Extraction Procedure (FIG. **1**)

FIG. **1** illustrates in terms of functional blocks a procedure, hereinafter referred to as "extraction procedure" and identified with the reference **100**, directed to process an input image in order to obtain an optimal set of keypoints and generate a corresponding set of descriptors according to an embodiment of the present invention. The keypoints and the descriptors will be then exploited for image analysis purposes. In the following of the present description, the generic expressions "image analysis" and "analyzing an image" have to be intended to comprise all those operations which provide for comparing an image with at least one another image. These operations may be carried out in a wide variety of applications, such as for example in an object recognition application, as well as in an application providing for the creation of a single panoramic picture starting from a plurality of different pictures.

As will be described later on, the extraction procedures according to an embodiment of the present invention further provides for selecting an optimal subset of keypoints and compressing the descriptors of such keypoints to an extent such to greatly improve the efficiency of subsequent procedures.

The steps of the extraction procedure **100** described in this section may be carried out by proper processing units, whose structure and function depends on the specific field of application to which they are destined. For example, each processing unit may be a hardware unit specifically designed to perform one or more steps of the method. Moreover, the steps of the method may be carried out by a programmable machine (e.g., a computer) under the control of a corresponding set of instructions.

Keypoints Extraction (Phase **110**)

The first phase **110** of the extraction procedure **100** provides for receiving a query image **115** and extracting therefrom a first set of keypoints KP, each one associated with a corresponding pair of spatial coordinates C identifying the location of such keypoint KP within the query image **115**.

This operation may be carried out by exploiting the known Difference of Gaussians (DoG) keypoint extraction algorithm; however, similar considerations apply in case different keypoint extraction algorithms are employed, such as for example the Determinant of the Hessians (DoH) keypoint extraction algorithm. Making reference to the DoG keypoint extraction algorithm, the query image **115** is convolved with Gaussian filters in a sequence at different scales. Then, a

difference operation is carried out between pairs of adjacent Gaussian-blurred images in the sequence. The keypoints KP are then chosen as the points having maximum/minimum values of Difference of Gaussian (DoG) at multiple scales. Particularly, each pixel in a DoG image is compared to its eight neighbors at the same scale and to nine neighboring pixels at each of the neighboring scales (i.e., the subsequent and the previous scales in the sequence). If the pixel value is the maximum or minimum among all compared pixels, that point is considered a candidate keypoint KP.

The phase **110** also provides that each keypoint KP is assigned to one or more orientations based on local image luminance gradient directions. For example, an orientation histogram with a plurality of bins is formed, with each bin covering a corresponding degree interval. Each sample in the neighboring window added to a histogram bin is weighted by its gradient magnitude and by a Gaussian-weighted circular window. The peaks in the resulting histogram correspond to dominant orientations. Once the histogram is filled, the orientations corresponding to the highest peak and local peaks that are within 80% of the highest peaks are assigned to the keypoint KP. In case of multiple orientations have been assigned, an additional keypoint KP is created having the same location and scale as the original keypoint for each additional orientation.

At the end of phase **110** a set of keypoints KP is thus generated, together with the corresponding coordinates C, the scale S at which the keypoint is extracted, its dominant orientation O, and the peak P, i.e., the absolute value of the DoG corresponding to such keypoint (which is indicative of the contrast thereof).

Descriptors Generation (Phase **120**)

The following phase **120** provides to process the query image **115** in order to compute for each keypoint KP a corresponding descriptor D. In the example at issue, the descriptors D computed at phase **120** are descriptor of the SIFT type. While the keypoints KP have been extracted in such a way to ensure invariance to image location, scale and rotation, the SIFT descriptors D are computed in such a way to be highly distinctive and partially invariant to illumination and viewpoint. Specifically, for each keypoint KP a set of 16 sub-histograms are calculated on a 4×4 grid that is centered at the keypoint KP location and orientated according to the dominant orientation of the keypoint KP. Each sub-histogram includes 8 bins, each one corresponding to an orientation having an angle $n*\pi/4$ (n=0, 1, . . . 7) with respect to the dominant orientation; the frequency of each bin of a sub-histogram is proportional to the luminance gradient of the grid cell (hereinafter referred to as sub-region) corresponding to such sub-histogram, considered along the direction identified by such bin. The values of such orientation histograms are arranged in an array, forming the descriptor D of the keypoint KP. Since there are 4×4=16 sub-histograms each with 8 bins, the descriptor D is an array having 128 items.

The concepts of the present invention are also applicable if the SIFT descriptor is calculated on a grid including a different number of cells, and/or with a different number of bins per histogram.

Moreover, even if in the example at issue reference has been made to descriptors of the SIFT type, similar considerations apply in case different types of descriptors are employed, such as for example the Speeded Up Robust Feature (SURF) and the Histogram of Oriented Gradients (HOG), or possibly others. Furthermore, even if reference has been made and will be made in the following to descriptors comprising data relating to luminance gradients, similar considerations apply if gradients of different parameters are considered. Indeed, as it is well known to those skilled in the art, the luminance is only one of the physical properties of the color. Thus, even if the luminance has been ascertained to be the best (i.e., the most robust) physical property to be considered for image analysis purposes, different types of descriptors may be also considered, for example comprising data relating to chrominance gradients, saturation gradients, or even color (which includes both luminance, saturation and chrominance) gradients.

As already mentioned above, carrying out image analysis operations involves the management of a quite large amount of data: indeed, each keypoint KP is associated with a plurality of local features (hereinafter globally identified with reference LFkp), including the coordinates C, the scale S, the dominant orientation O, and the peak P, as well as a corresponding descriptor D formed by an array of 128 items. For this purpose, in order to reduce the overall amount of data to be managed (e.g., to be memorized and/or transmitted), the extraction procedure **100** according to an embodiment of the present invention provides for two expedients, i.e.:

1) reducing the number of the previously generated keypoints KP by selecting the most relevant keypoints KP (from the image comparison point of view), in order to obtain an optimal subset SUB of keypoints KP, and
2) properly compressing both the coordinates C and the descriptors D.

Phase **130** of the extraction procedure **100** is dedicated to the selection of the optimal subset SUB, phase **140** is dedicated to the compression of the descriptors D, and phase **150** is dedicated to the compression of the coordinates C.

Selection of the Optimal Subset of Keypoints (Phase **130**)

According to an embodiment of the present invention, the selection of the optimal subset SUB is carried out by calculating for at least one local feature LFkp—the coordinates C, the scale S, the dominant orientation O, the peak P and the descriptor D—of each keypoint KP of the query image **115** at least one corresponding feature relevance probability FRP, sorting the keypoints KP according to a keypoint relevance probability KRP based on the feature relevance probabilities FRP of its local features LFkp, and then selecting the keypoints KP having the highest keypoint relevance probabilities KRP.

According to an embodiment of the present invention, the feature relevance probability FRP of each local feature LFkp of the generic keypoint KP is calculated by exploiting a corresponding reference statistical distribution Rsd, which has been already predetermined in advance after having carried out statistical evaluations on a benchmark image database.

The reference statistical distributions Rsd are made in such a way to reflect the statistical behavior of the local features LFkp of keypoints KP considered useful for image analysis purposes.

For example, in case of object recognition procedures, the benchmark image database is a database comprising a plurality of image pairs, with each image pair consisting of two pictures depicting a same object/scene. According to an embodiment of the present invention, the reference statistical distributions are generated in the following way.

Keypoints are firstly extracted from all the images of the benchmark database. Then, a first statistical analysis is carried out on one or more selected local features of all the extracted keypoints, so as to generate first statistical distributions of such selected local features. Each first statistical distribution of a local feature is arranged in the form of a histogram, obtained by counting the number of keypoints (keypoints frequency)—among the totality of keypoints

extracted from the images of the benchmark database—having a value of such local feature that falls within each of a plurality predefined local feature value intervals (bin). Then, for each image pair, keypoints of one picture are matched with keypoints of the other picture. The matches among such keypoints are processed using an image comparison procedure (such as any one among the known image comparison procedures based on image feature matching) in order to identify which match is correct (inlier) and which is incorrect (outlier). A second statistical analysis is then carried out on the same feature or features previously considered in order to generate the reference statistical distributions Rsd to be used for calculating the feature relevance probabilities FRP. This time, the generation of the reference statistical distributions Rsd is carried out by calculating for each bin a ratio between the number of keypoints belonging to inliers and having a value of the corresponding local feature that falls within said bin, and the total number of keypoints (both belonging to inliers and outliers) having a value of the corresponding local feature that falls within the same bin. The Applicant has observed that the first statistical distributions and the reference statistical distributions Rsd are quite different to each other. Since the reference statistical distributions Rsd are generated taking into account the keypoints that involve a correct feature match (inlier), the Applicant has found that such statistical distributions are good representatives of the statistical behavior of keypoints (hereinafter, "relevant keypoints") which are relevant for image analysis purposes, and particularly suited for being efficiently employed in an image comparison procedure.

FIGS. **2**A-**2**F illustrate some statistical distributions Rsd of corresponding selected local features LFkp of keypoints KP according to some exemplary embodiments of the present invention. In particular, the statistical distributions Rsd of FIGS. **2**A-**2**F have been generated from images of a benchmark database specifically arranged for object recognition applications. Should a different image analysis application be considered, such as for example the creation of a single panoramic picture starting from a plurality of different pictures, the images of the benchmark, and therefore, the resulting statistical distributions Rsd would be different.

FIG. **2**A is a statistical distribution Rsd related to the coordinates C of the keypoints KP. Each bin of the corresponding histogram represents the distance (in pixel) of the generic keypoint KP from the center of the image. In the example at issue, the considered image is of the VGA type (i.e., having a resolution of 640×480), thus the center corresponds to the coordinate (320, 240). According to the histogram illustrated in FIG. **2**A, the bin having the highest keypoints KP frequency is the one corresponding to the center of the image. This means that the closer a keypoint KP is to the center, the higher the probability that such keypoint KP is a relevant keypoint; the trend of the histogram frequencies monotonically decreases as the distance from the center increases. This could be easily explained by the fact that when an object is photographed, it is highly probable that said object is framed in the center of the picture. It has to be appreciated that in this case the bins of the histogram do not have all the same widths; this is due to the fact that the width of each bin has been properly determined by a (scalar and/or vector) quantizer in such a way to compute few bins, avoiding thus the occurrence of overfitting phenomenon occurrences. The concepts of the present invention also apply in case a (scalar and/or vector) uniform quantization is employed, i.e., with all the bins of the histogram that have a same width.

FIG. **2**B is a statistical distribution Rsd related to the dominant orientation O of the keypoints KP. Each bin of the corresponding histogram represents the angle (in radians) of the dominant direction of the generic keypoint KP with respect to the horizon (corresponding to 0 radians). According to the histogram illustrated in FIG. **2**B, the bins having the highest keypoints KP frequencies are the ones corresponding to the orientations which are parallel or perpendicular to the horizon orientation (i.e., corresponding to $\pi/2$, 0, $-\pi/2$, $-\pi$). This means that the closer the orientation of a keypoint KP is to one of said orientations, the higher the probability that such keypoint KP is a relevant keypoint. This could be explained by the fact that when an object is photographed, it is highly probable that said object is framed so as to mainly extend parallel and/or perpendicular to the horizon line. In this case as well, the width of the bins is determined by means of a quantizer.

FIG. **2**C is a statistical distribution Rsd related to the peak P of the keypoints KP. Each bin of the corresponding histogram represents the contrast between the generic keypoint KP and the most similar point among the neighbor ones. According to the histogram illustrated in FIG. **2**C, the bin having the highest keypoints KP frequency is the one corresponding to the highest peak values. This means that the higher the contrast of a keypoint KP, the higher the probability that such keypoint KP is a relevant keypoint; the trend of the histogram frequencies monotonically increases as the contrast increases. This could be easily explained by the fact that a point of a picture having a high contrast is easily recognizable and identifiable. In this case as well, the width of the bins is determined by means of a quantizer.

FIG. **2**D is a statistical distribution Rsd related to the scale S of the keypoints KP. Each bin of the corresponding histogram represents a particular scale S at which the keypoint KP may be extracted. According to the histogram illustrated in FIG. **2**D, the bin having the highest keypoints KP frequency corresponds to a mid-low scale. In this case as well, the width of the bins is determined by means of a quantizer.

FIG. **2**E is a first statistical distribution Rsd related to the descriptors D of the keypoints KP. In this case, the corresponding histogram is three-dimensional, with each bin thereof corresponding to interval values of two parameters of the descriptor D of the generic keypoint KP, i.e., the mean (x axis) and the variance (y axis) of the descriptor D. Greater frequency values are indicated by circles of larger diameter. The mean and the variance have been considered together to form a same histogram, since they are linked to each other. According to such histogram, the bin having the highest keypoints KP frequency, represented by larger circles, is the one corresponding to the highest mean and the lowest variance. This can be explained by the fact that the higher the mean of the descriptor D of a keypoint KP, the higher the luminance gradient corresponding to such keypoint KP, and the lower the variance of the descriptor D of a keypoint KP, the lower the unwanted noise affecting such keypoint KP.

FIG. **2**F is a second statistical distribution Rsd related to the descriptors D of the keypoints KP. In this case, each bin corresponds to a particular maximum distance between the descriptor D of a keypoint KP and the descriptors D of the other keypoints KP of the same image. For example, such maximum distance may be computed based on the Euclidean distance between descriptors, Other known method may be also contemplated, such as for example exploiting the symmetrized Kullback-Leibler divergence.

Returning to FIG. **1**, according to an embodiment of the present invention, phase **130** of the extraction procedure **100** provides for calculating, for each keypoint KP extracted at phase **110**:

A first feature relevance probability FRP**1**, obtained from the statistical distribution Rsd related to the coordinates C of said keypoint KP. The histogram corresponding to said distribution is inspected in order to identify the bin thereof fitting the coordinates C of said keypoint KP; then, the feature relevance probability FRP**1** is set equal to the keypoints frequency of the identified bin.

A second feature relevance probability FRP**2**, obtained from the statistical distribution Rsd related to the dominant orientation O of said keypoint KP. The histogram corresponding to said distribution is inspected in order to identify the bin thereof fitting the dominant orientation O of said keypoint KP; then, the feature relevance probability FRP**2** is set equal to the keypoints frequency of the identified bin.

A third feature relevance probability FRP**3**, obtained from the statistical distribution Rsd related to the peak P of said keypoint KP. The histogram corresponding to said distribution is inspected in order to identify the bin thereof fitting the peak P of said keypoint KP; then, the feature relevance probability FRP**3** is set equal to the keypoints frequency of the identified bin.

A fourth feature relevance probability FRP**4**, obtained from the statistical distribution Rsd related to the scale S of said keypoint KP. The histogram corresponding to said distribution is inspected in order to identify the bin thereof fitting the scale S of said keypoint KP; then, the feature relevance probability FRP**4** is set equal to the keypoints frequency of the identified bin.

A fifth feature relevance probability FRP**5**, obtained from the statistical distribution Rsd related to the mean and the variance of the descriptor D of said keypoint KP. The histogram corresponding to said distribution is inspected in order to identify the bin thereof fitting the mean and the variance of the elements of the descriptor D of said keypoint KP; then, the feature relevance probability FRP**5** is set equal to the keypoints frequency of the identified bin.

A sixth feature relevance probability FRP**6**, obtained from the statistical distribution Rsd related to the maximum distance (e.g., the Euclidean distance) between the descriptor D of said keypoint KP and the descriptors D of the other keypoints KP. The histogram corresponding to said distribution is inspected in order to identify the bin thereof fitting such distance; then, the feature relevance probability FRP**6** is set equal to the keypoints frequency of the identified bin.

Therefore, for each keypoint KP, a keypoint relevance probability KRP is obtained by at least one of, or by combining among them the feature relevance probabilities FRP of the local features thereof. For example, starting with the assumption that the feature relevance probabilities FRP are independent to one another, the keypoint relevance probability KRP of the generic keypoint KP is calculated by multiplying to each other its corresponding feature relevance probabilities FRP. Generally, the higher the number of different feature relevance probabilities FRP used to calculate the keypoint relevance probability KRP, the better the results obtainable by employing such method. By considering the example of SIFT descriptors for visual searching applications, it is preferable that the feature relevance probabilities considered for calculating the keypoint relevance probability include at least those corresponding to the scale, the peak and the distance from the centre.

FIG. **2**G is an exemplary picture in which a plurality of keypoints are identified by means of corresponding circular spots, each one having a diameter that is proportional to the relevance probability KRP of the keypoint.

Once the keypoint relevance probabilities KRP of all the keypoints KP extracted in phase **110** have been calculated, said keypoints KP are sorted in a sequence according to a decreasing keypoint relevance probability KRP order. Then, the optimal subset SUB is formed by taking a number (based on the desired reduction in the amount of data to be managed) of keypoints KP from the first ones of the ordered sequence. The selected keypoints KP belonging to the optimal subset SUB results to be the most relevant keypoints KP (from the image comparison point of view) among the totality of keypoints KP extracted in phase **110**. In this way, the reduction of the overall amount of data is carried out in a smart and efficient way, taking into account only the relevant keypoints KP, and discarding those that are less useful.

It is underlined that although the selection of the optimal subset of keypoints according to the embodiment of the invention above described provides for calculating each feature relevancy probability exploiting a corresponding statistical distribution Rsd obtained by calculating for each bin thereof a ratio between the keypoint inliers having a value of the corresponding local feature that falls within said bin, and the total number of keypoints having a value of the corresponding local feature that falls within the same bin, the concepts of the present invention are also applicable in case different, statistically equivalent statistical distributions are employed, obtained with different, even manual, methods. In the following description, two statistical distributions are considered statistically equivalent one to another if they allow to obtain similar feature relevancy probabilities starting from a same set of keypoints.

Compression of the Descriptors (Phase **140**)

According to an embodiment of the present invention, the compression of the descriptors D is carried out through vector quantization, by exploiting a reduced number of optimized codebooks.

FIG. **3**A illustrates an exemplary descriptor D of the SIFT type (one of the descriptors D generated at phase **120** of the extraction procedure **100** of FIG. **1** which has been selected to be part of the optimal subset SUB) corresponding to a generic keypoint KP. As already mentioned above, the descriptor D comprises sixteen sub-histograms shi (i=1, 2, . . . , 16), each one showing how the luminance gradient of a respective sub-region of the image close to the keypoint KP is distributed along eight directions. Specifically, each sub-histogram shi is associated with a sub-region corresponding to one of 16 cells of a 4×4 grid that is centered at the keypoint KP location and oriented according to the dominant orientation O of the keypoint KP; each sub-histogram shi includes eight bins, each one corresponding to an orientation having an angle $n*\pi/4$ (n=0, 1, . . . 7) with respect to the dominant orientation O.

As illustrated in FIG. **3**B, the values of all the orientation histograms shi of a descriptor D are arranged in a corresponding descriptor array, identified in figure with the reference DA. The descriptor array DA comprises sixteen elements ai (i=1, 2, . . . , 16), each one storing the values taken by a corresponding sub-histogram shi (i=1, 2, . . . , 16); each element ai comprises in turn eight respective items, each one storing a frequency value corresponding to a respective one of the eight bins of the sub-histogram shi. Thus, each descriptor array DA includes 16*8=128 items. By considering that in a SIFT descriptor D a typical frequency value may range from 0 to 255, each item of the descriptor array DA can be represented with a byte; therefore, the memory occupation of the descriptor array DA is equal to 128 bytes. Thus, making reference again to the extraction procedure **100** of FIG. **1**, the

amount of data (in bytes) corresponding to all the descriptors D of the keypoints KP belonging to the selected optimal subset SUB is equal to 128 multiplied by the number of keypoints KP of the optimal subset SUB.

In order to reduce this amount of data, the descriptor arrays DA corresponding to such descriptors D are compressed through vector quantization.

As it is well known to those skilled in the art, compressing a data array formed by n elements (n-tuple) by exploiting vector quantization provides for jointly quantizing the set of all the possible n-tuple values which the data array may assume into a reduced set comprising a lower number of n-tuple values (which values may even differ from the values of the set to be quantized). Since the reduced set comprises a lower number of n-tuple values, it requires less storage space. The n-tuple values forming the reduced set are also referred to as "codewords". Each codeword is associated with a corresponding set of different n-tuple values the array may assume. The association relationships between n-tuple values of the data array and codewords is determined by means of a corresponding codebook.

Making reference in particular to the descriptor array DA, which includes 16 elements ai formed in turn by eight items each having values ranging from 0 to 255, the descriptor array DA may take a number $N=256^{128}$ of different 16-tuple values. By applying compression through vector quantization, such N different 16-tuple values are approximated with a number N1<N of codewords of a codebook. The codebook determines association relationships between each codeword and a corresponding set of 16-tuple values of the descriptor array DA. Therefore, each codeword of the codebook is a 16-tuple value which is used to "approximate" a corresponding set of 16-tuple values of the descriptor array DA. The vector quantization is a lossy data compression, whose accuracy can be measured through a parameter called distortion. The distortion may be for example calculated as the Euclidean distance between a generic codeword of the codebook and the set of n-tuple values of the array which are approximated by such codeword. Similar considerations apply even if the distortion is calculated with a different method. In any case, broadly speaking, the higher the number N1 of codewords of a codebook, the lower the distortion of the compression.

As it is well known to those skilled in the art, the generation of the codewords of a codebook is typically carried out by performing statistical operations (referred to as training operations) on a training database including a collection of a very high number of training arrays. Making reference in particular to the descriptor array DA, the training database may include several millions of training descriptor arrays, wherein each training descriptor array is one of the $N=256^{128}$ possible 16-tuple values the descriptor array DA may assume.

According to a solution illustrated in FIG. **4**A, the whole descriptor array DA is compressed using a single codebook CBK comprising N1 16-tuple value codewords CWj (j=1, 2, . . . N1). Therefore, with N1 different codewords CWj, the minimum number of bits required to identify the codewords is equal to $\log_2$ N1. As already mentioned above, the generation of the N1 different codewords CWj of such single codebook CBK is carried out by performing training operations on a plurality of training descriptor arrays, wherein each training descriptor array is one of the $N=256^{128}$ possible 16-tuple values the descriptor array DA may assume.

In order to keep the compression distortion under a sufficiently reduced threshold such as not to impair the outcome of the subsequent image analysis operations, the required codewords number N1 may become very high. Having a codebook formed by too high a number N1 of codewords is disadvantageous under different points of view. Indeed, the number of training arrays to be used for generating the codewords would become excessive, and the processing times would become too long. Moreover, in order to carry out compression operations by exploiting a codebook, the whole N1 codewords forming the latter have to be memorized somewhere, occupying a non-negligible amount of memory space. The latter drawback is quite critical, since the hardware employed for image analysis applications (e.g., Graphic Processing Units, GPU) may be equipped with not so capacious memories.

Making reference to FIG. **4**B, in order to reduce the whole number of codewords CWj to be managed without increasing the distorsion, the descriptor array DA may be subdivided into a plurality of sub-arrays SDAk (k=1, 2, . . .), each one comprising a respective number mk of elements ai of the descriptor array DA, and then each sub-array SDAk is individually compressed using a respective codebook CBKk comprising N2 mk-tuple value codewords CWj (j=1, 2, . . . N2).

In the example illustrated in FIG. **4**B, the descriptor array DA is subdivided into four sub-arrays SDAk (k=1, 2, 3, 4), each one comprising mk=4 elements ai of the descriptor array DA:

- the first sub-array SDA**1** is formed by the element sequence a**1**, a**2**, a**3**, a**4**;
- the second sub-array SDA**2** is formed by the element sequence a**5**, a**6**, a**7**, a**8**;
- the third sub-array SDA**3** is formed by the element sequence a**9**, a**10**, a**11**, a**12**, and
- the fourth sub-array SDA**4** is formed by the element sequence a**13**, a**14**, a**15**, a**16**.

The compression of each sub-array SDAk is carried out using a respective codebook CBKy (y=k) comprising N2 4-tuple value codewords CWj (j=1, 2, . . . N2). Therefore, with 4*N2 different codewords CWj, the minimum number of bits required to identify all the codewords is equal to $4*\log_2$ N2. Even if in the considered case each sub-array SDAk has been compressed using a codebook CBKy comprising a same number N2 of codewords CWj, similar considerations apply in case each sub-array SDAk is compressed using a respective, different, number of codewords CWj.

In the case illustrated in FIG. **4**B, the generation of the N2 different codewords CWj of each codebook CBKy is carried out by performing training operations on a respective sub-set of training descriptor arrays. Each sub-set of training descriptor arrays of a codebook CBKk corresponds to one of the four sub-arrays SDAk, and may be obtained by considering from each training descriptor array used to generate the single codebook CBK of FIG. **4**A only the portion thereof corresponding to the sub-array SDAk. For example, in order to generate the codebook CBK**1**, only the first four elements a**1**, a**2**, a**3**, a**4** of the 16-tuple training descriptor arrays used to generate the single codebook CBK of FIG. **4**A are employed.

Compared to the case of FIG. **4**A, in which the whole descriptor array DA is compressed using a single codebook CBK formed by codewords CWj having the same dimension of the descriptor array DA itself (16 elements), the use of codebooks CBKy formed by codewords CWj having a (smaller) dimension mk of a sub-array SDAk thereof (e.g., mk=4 elements) allows to obtain, with a same number of codewords CWj, a lower distortion.

Having fixed the total number of codewords CWj, the higher the number of sub-arrays SDAk which the descriptor array DA is subdivided in, the lower the distortion, but—at the same time—the higher the minimum number of bits required to identify all the codewords CWj.

According to an embodiment of the present invention, the subdivision of the descriptor array DA in sub-arrays SDAk for compression purposes is carried out by taking into consideration the occurrence of correlation relationships among the elements ai of the descriptor array DA.

As already described with reference to FIGS. **3**A and **3**B, each element ai of the descriptor array DA stores the values taken by the sub-histogram shi associated with a respective sub-region, which sub-region corresponds in turn to a cell of the 4×4 grid centered at the keypoint KP corresponding to such descriptor array DA.

According to an embodiment of the present invention illustrated in FIG. **5**, after having carried out statistical behavioral analysis on a large amount of descriptor arrays DA (for example exploiting the training descriptor arrays of the training database), it has been found that the sub-histograms shi of a generic keypoint KP can be arranged in correlation families CFx (x=1, 2, 3, 4), with each correlation family CFx comprising a set of correlated sub-histograms shi with a similar statistical behavior, i.e., with a similar trend of the bin frequencies. For example, two sub-histograms shi belonging to a same correlation family CFx may have a similar number of frequency peaks at same (or similar) bins.

The statistical behavioral analysis employed to form the correlation families CFx showed that, having fixed the maximum number of codewords CWj to be used for compressing the descriptor array DA, if the arrangement of the sub-histograms shi in correlation families CFx is varied (by assigning the sub-histograms shi to different correlation families CFx), the resulting distortion accordingly varies. The correlation families CFx are thus formed by considering, among all the possible sub-histograms shi subdivisions, the one corresponding to the lowest distortion.

After having performed such statistical behavioral analysis it has also been found that the correlation between the statistical behavior of two sub-histograms shi depends on two main parameters, i.e., the distance of the sub-regions associated to the sub-histograms shi from the keypoint KP and the dominant orientation thereof.

Making reference to FIG. **5**, the sixteen sub-histograms shi of a keypoint KP are arranged in four correlation families, i.e.:

- a first correlation family CF**1** comprising the sub-histograms sh**1**, sh**4**, sh**13** and sh**16**;
- a second correlation family CF**2** comprising the sub-histograms sh**2**, sh**3**, sh**14** and sh**15**;
- a third correlation family CF**3** comprising the sub-histograms sh**5**, sh**8**, sh**9** and sh**12**, and
- a fourth correlation family CF**4** comprising the sub-histograms sh**6**, sh**7**, sh**10** and sh**11**.

According to an embodiment of the present invention, the above identified correlation families CFx are advantageously exploited in order to compress the descriptor array DA using a reduced number of optimized codebooks CBKy. The subdivision of the descriptor array DA in sub-arrays SDAk is carried out in such a way that at least two sub-arrays SDAk have the same global (i.e., considering all the elements thereof) statistical behavior; in this way, it is possible to use a single codebook CBKy to compress more than one sub-arrays SDAk. For this purpose, the subdivision of the descriptor array DA is carried out in such a way to obtain group(s) of sub-arrays SDAk in which for each group the elements ai occupying the same position in all the sub-arrays SDAk of the group belong to a same correlation family CFx. Therefore, all the sub-arrays SDAk belonging to a same group can be advantageously compressed using a same corresponding codebook CBKy, whose codewords CWj are obtained by considering, from each training descriptor array used to generate the single codebook CBK of FIG. **4**A, only the elements thereof belonging to the correlation families CFx which the elements ai of the sub-arrays SDAk of the group belong to.

According to an exemplary embodiment of the present invention illustrated in FIG. **6**A, the descriptor array DA is subdivided in four sub-arrays SDA**1**-SDA**4** which are arranged in a single group. Therefore, all the sub-arrays SDAk are compressed using a same codebook CBK**1**. Specifically:

- the first sub-array SDA**1** is formed by the element sequence a**1**, a**2**, a**6**, a**5**;
- the second sub-array SDA**2** is formed by the element sequence a**4**, a**3**, a**7**, a**8**;
- the third sub-array SDA**3** is formed by the element sequence a**16**, a**15**, a**11**, a**12**, and
- the fourth sub-array SDA**4** is formed by the element sequence a**13**, a**14**, a**10**, a**9**.

In this case:

- the first elements ai of each sub-array SDAk belong to the first correlation family CF**1**;
- the second elements ai of each sub-array SDAk belong to the second correlation family CF**2**;
- the third elements ai of each sub-array SDAk belong to the fourth correlation family CF**4**, and
- the fourth elements ai of each sub-array SDAk belong to the third correlation family CF**3**.

The codebook CBK**1** for compressing the generic sub-array SDA**1**-SDA**4** includes N**3** codewords CWj, wherein each codeword CWj has the first element belonging to the first correlation family CF**1**, the second element belonging to the second correlation family CF**2**, the third element belonging to the fourth correlation family CF**4**, and the fourth element belonging to the third correlation family CF**3**.

With N**3** different codewords CWj, the minimum number of bits required to identify all the codewords is equal to $4*(\log_2 N3)$.

According to another exemplary embodiment of the present invention illustrated in FIG. **6**B, the descriptor array DA is subdivided in two sub-arrays SDA**1**, SDA**2** which are arranged in a single group. Therefore, all the sub-array SDAk are compressed using a same codebook CBK**1**. Specifically:

- the first sub-array SDA**1** is formed by the element sequence a**1**, a**2**, a**3**, a**4**, a**5**, a**6**, a**7**, a**8**, and
- the second sub-array SDA**2** is formed by the element sequence a**13**, a**14**, a**15**, a**16**, a**9**, a**10**, a**11**, a**12**.

In this case:

- the first and the fourth elements ai of each sub-array SDAk belong to the first correlation family CF**1**;
- the second and the third elements ai of each sub-array SDAk belong to the second correlation family CF**2**;
- the fifth and the eighth elements ai of each sub-array SDAk belong to the third correlation family CF**3**, and
- the sixth and the seventh elements ai of each sub-array SDAk belong to the fourth correlation family CF**4**.

The codebook CBK**1** for compressing the generic sub-array SDA**1**, SDA**2** includes N**4** codewords CWj, wherein each codeword CWj has the first and the fourth elements belonging to the first correlation family CF**1**, the second and the third elements belonging to the second correlation family CF**2**, the fifth and the eighth elements belonging to the third correlation family CF**3**, and the sixth and the seventh elements belonging to the third correlation family CF**3**.

With N**4** different codewords CWj, the minimum number of bits required to identify all the codewords is equal to $2*(\log_2 N4)$.

According to another exemplary embodiment of the present invention illustrated in FIG. **6**C, the descriptor array

DA is subdivided in six sub-arrays SDA1-SDA6, four of which (SDA1-SDA4) are arranged in a first group, and two of which (SDA5, SDA6) are arranged in a second group. Therefore, the sub-arrays SDA1-SDA4 are compressed using a same first codebook CBK1, while the sub-arrays SDA5-SDA6 are compressed using a same second codebook CBK2. Specifically:

- the first sub-array SDA1 is formed by the element sequence a5, a1, a2;
- the second sub-array SDA2 is formed by the element sequence a8, a4, a3;
- the third sub-array SDA3 is formed by the element sequence a9, a13, a14;
- the fourth sub-array SDA4 is formed by the element sequence a12, a16, a15;
- the fifth sub-array SDA5 is formed by the element sequence a6, a7, and the sixth sub-array SDA6 is formed by the element sequence a10, all.

In this case:

- the first elements ai of each sub-array SDA1-SDA4 of the first group belong to the third correlation family CF3;
- the second elements ai of each sub-array SDA1-SDA4 of the first group belong to the first correlation family CF1;
- the third elements ai of each sub-array SDA1-SDA4 of the first group belong to the second correlation family CF2, and
- the first and second elements ai of each sub-array SDA5-SDA6 of the second group belong to the fourth correlation family CF4.

The codebook CBK1 for compressing the generic sub-array SDA1-SDA4 belonging to the first group includes N5 codewords CWj, wherein each codeword CWj has the first element belonging to the third correlation family CF3, the second element belonging to the first correlation family CF1, and the third element belonging to the second correlation family CF2. The codebook CBK2 for compressing the generic sub-array SDA5-SDA6 belonging to the second group includes N6 codewords CWj, wherein each codeword CWj has the first and second elements belonging to the fourth correlation family CF4.

With N5+N6 different codewords CWj, the minimum number of bits required to identify all the codewords is equal to $4*(\log_2 N5)+2*(\log_e N6)$.

According to another exemplary embodiment of the present invention illustrated in FIG. 6D, the descriptor array DA is subdivided in eight sub-arrays SDA1-SDA5, four of which (SDA1-SDA4) are arranged in a first group, and four of which (SDA5 -SDA8) are arranged in a second group. Therefore, the sub-arrays SDA1-SDA4 are compressed using a same first codebook CBK1, while the sub-arrays SDA5-SDA8 are compressed using a same second codebook CBK2. Specifically:

- the first sub-array SDA1 is formed by the element sequence a5, al;
- the second sub-array SDA2 is formed by the element sequence a8, a4;
- the third sub-array SDA3 is formed by the element sequence a9, a13;
- the fourth sub-array SDA4 is formed by the element sequence a12, a16;
- the fifth sub-array SDA5 is formed by the element sequence a6, a2;
- the sixth sub-array SDA6 is formed by the element sequence a7, a3;
- the seventh sub-array SDA7 is formed by the element sequence a10, a14, and
- the eighth sub-array SDA8 is formed by the element sequence a11, a15.

In this case:

- the first elements ai of each sub-array SDA1-SDA4 of the first group belong to the third correlation family CF3;
- the second elements ai of each sub-array SDA1-SDA4 of the first group belong to the first correlation family CF1;
- the first elements ai of each sub-array SDA5-SDA8 of the second group belong to the fourth correlation family CF4, and
- the second elements ai of each sub-array SDA5-SDA8 of the second group belong to the second correlation family CF2.

The codebook CBK1 for compressing the generic sub-array SDA1-SDA4 belonging to the first group includes N7 codewords CWj, wherein each codeword CWj has the first element belonging to the third correlation family CF3, and the second element belonging to the first correlation family CF1. The codebook CBK2 for compressing the generic sub-array SDA5-SDA8 belonging to the second group includes N8 codewords CWj, wherein each codeword CWj has the first elements belonging to the fourth correlation family CF4 and the second elements belonging to the second correlation family CF2.

Therefore, with N7+N8 different codewords CWj, the minimum number of bits required to identify all the codewords is equal to $4*(\log_2 N7)+4*(\log_2 N8)$.

Naturally, the concepts of the present invention are also applicable with subdivisions into a different number of sub-arrays and/or with a different number of codebooks. Moreover, even if in the present description reference has been made to the compression of a SIF descriptor calculated on a grid including 4×4 cells with eight bins per histogram, similar consideration apply if the number of cells and/or the number of bins per histogram is different, as well as descriptors of other types are considered.

Compared to the known solutions, with a same compression distortion, the combined use of subdividing the descriptor array DA in sub-arrays SDAk and employing a same codebook CBKy for more than one sub-arrays SDAk allows to drastically reduce the memory space required to store the codebook(s) CBKy used to compress the descriptor array DA. This is a great advantage, since, as already mentioned above, the hardware employed for image analysis applications (e.g., Graphic Processing Units, GPU) may be equipped with not so capacious memories. Another advantage given by the combined use of subdividing the descriptor array DA in sub-arrays SDAk and employing a same codebook CBKy for more than one sub-arrays SDAk consists in that the training procedure for the generation of the codebook(s) CBKy results to be faster.

The compression operations carried out in phase **140** of the extraction procedure **100** (see FIG. **1**) on each received descriptor D generate as a result a corresponding compressed descriptor array CDA, which approximate the value taken by the respective descriptor array DA. More specifically, for each codebook CBKy used to compress the descriptor array DA, each codeword CWj of such codebook CBKy is identified by a corresponding compression index Cy; if the codebook CBKy is formed by a number N of different codewords CWj, the compression index Cy is formed by at least $\log_2 N$ bits. For a descriptor array DA which has been subdivided into a set of sub-arrays SDAk, the corresponding compressed descriptor array CDA comprises a compression index Cy for each sub-array SDAk of the set, wherein each compression index Cy identifies the codeword CWj of the codebook CBKy used to approximate said sub-array SDAk.

Compression of the Coordinates (Phase **150**)

According to an embodiment of the present invention, the amount of data to be managed (e.g., to be memorized and/or transmitted) for performing image analysis operations is further reduced by compressing the coordinates C of the keypoints KP belonging to the optimal subset SUB calculated at phase **130** of the extraction procedure **100** (see FIG. **1**).

FIG. **7**A illustrates an exemplary distribution of the keypoints KP of the optimal subset SUB within a bi-dimensional space corresponding to the query image **115**; each keypoint KP is associated with a corresponding pair of spatial coordinates C identifying the location of such keypoint KP within the query image **115**.

Firstly, the coordinates C of all the keypoints KP of the subset SUB are quantized. For this purpose, a n×m grid is superimposed over the query image **115**. In the example illustrated in FIG. **7**B, the grid has n=10 rows and m=15 columns.

A bi-dimensional histogram is then generated by counting for each cell of the grid (corresponding to a bin of the histogram) the number of keypoints KP which lie therewithin. FIG. **7**C is an exemplary graphical depiction of the histogram obtained by superimposing the grid of FIG. **7**B over the set of keypoints KP of FIG. **7**A. In the graphical depiction of FIG. **7**C, the cells void of keypoints KP are colored in black, while the cells including at least a keypoint KP are colored in gray. In the example at issue (wherein the cells including the highest number of keypoints include two keypoints), the cells including a single keypoint KP are colored in dark grey, while those including two keypoints KP are colored in a lighter grey.

The histogram obtained from the keypoint counting has a great number of bins whose frequency is equal to zero, i.e., with the corresponding cell that does not include any keypoint KP (the black cells depicted in FIG. **7**C).

The data representing the histogram may be advantageously compressed taking into considerations that the portions thereof corresponding to the zero frequency bins only provide the information that its corresponding cell does not include any keypoint.

For this purpose, the rows and the columns of the grid which are entirely formed by cells that does not include any keypoints KP can be advantageously removed. However, since the removal of such rows and/or columns would alter the absolute and relative positions of the keypoints KP, an indication of the positions of all the rows and columns void of keypoints KP (comprising those corresponding to the rows and/or columns to be removed) should be recorded.

For this purpose, two arrays r and c are defined in the following way:

- the array r is an array including an element for each row of the grid, wherein the generic element of the array is set to a first value (e.g., 0) if the corresponding cell of the grid does not include any keypoint KP, and it is set to a second value (e.g., 1) if the corresponding cell of the grid includes at least a keypoint KP, and
- the array c is an array including an element for each column of the grid, wherein the generic element of the array is set to a first value (e.g., 0) if the corresponding cell of the grid does not include any keypoint KP, and it is set to a second value (e.g., 1) if the corresponding cell of the grid includes at least a keypoint KP.

Once the arrays r and c have been generated, the next step provides for identify the rows and/or the columns which are entirely formed by cells that does not include any keypoints KP are identified. Making reference to the example at issue, such rows and columns are depicted in black in FIG. **7**D.

The rows and/or the columns of the grid which are entirely formed by cells that do not include any keypoints KP are then removed, and the resulting portions of the grid are compacted in order to fill the empty spaces left by the removals. Thus, in the resulting (compacted) grid, referred to as rank-**1** support, all the rows and all the columns include at least one cell comprising at least one keypoint KP. The histogram over the rank-**1** support corresponding to the example at issue is illustrated in FIG. **7**E.

From such histogram two different pieces of information can be extracted, i.e.:

1) the positions of the cells of the rank-**1** support including at least one keypoint KP, and
2) for each cell of the rank-**1** support identified at point **1**), the number of keypoints KP included therein.

Advantageously, as proposed by S. Tsai, D. Chen, G. Takacs, V. Chandrasekhar, J. P. Singh, and B. Girod in "Location coding for mobile image retrieval", *Proc. Int. Mobile Multimedia Conference* (*MobiMedia*), 2009, the information corresponding to point **1**) may be extracted exploiting a so-called "histogram map", while the information corresponding to point **2**) may be arranged in a so-called "histogram count".

The histogram map is a bi-dimensional mapping of the histogram over the rank-**1** support which identifies the bins thereof having a frequency equal to or higher than **1**. The histogram map corresponding to the histogram over the rank-**1** support of FIG. **7**E is illustrated in FIG. **7**F.

The histogram map can be represented with a corresponding matrix, whose generic element is equal to zero if the corresponding cell of the rank-**1** support does not include any keypoint KP, and is equal to one if the corresponding cell of the rank-**1** support does include at least one keypoint KP. The matrix of the histogram map illustrated in FIG. **7**F is the following one:

$$\begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

According to an embodiment of the present invention, the information provided by the histogram map can be advantageously compressed using an entropic coding optimized based on the statistical behavior of exemplary rank-**1** support histograms learned from the analysis of a large number of training images.

From such analysis it has been found that the locations of the keypoints KP within the generic image are such to entail a common statistical distribution of the "1" within the matrix of the histogram map.

The entropic coding is carried out in the following way.

The matrix of the histogram map is scanned (e.g., column by column) so as to subdivide it into a plurality of words each having a same length x. Based on the statistical analysis carried out on the training images, a word histogram is generated including a bin for each possible value the x-tuple of the generic word may take, with the frequency of each bin that indicates the probability that the x-tuple of the word takes the value associated with such bin. Briefly, such statistical analy-

sis has been carried out by making the assumption that the elements of the matrix of the histogram map are independent of each another. By analyzing a very high number of training images, it can be identified which is the probability that a "1" occurs in the matrix every n "0"; then, the word histogram is generated based on such probability.

FIG. **8**A illustrates an example of a word histogram in which the length x of the words is equal to six, and wherein each bin is identified by the decimal value of the corresponding x-tuple value. As expected, the highest frequency corresponds to the x-tuple (0,0,0,0,0,0), since there is a very higher probability that the generic cell of the rank-**1** support does not include any keypoint KP. The following highest probability is the one corresponding to a single keypoint KP for cell (x-tuple (1,0,0,0,0,0), (0,1,0,0,0,0), (0,0,1,0,0,0), (0,0,0,1,0,0), (0,0,0,0,1,0), (0,0,0,0,0,1)), then the one corresponding to two keypoints KP for cell, and so on.

The words are encoded with an entropic coding technique (e.g., the Huffman technique or the Arithmetic coding technique) by using for each word a coded word bci (i=1, 2, . . .) having a number of bits that depends on the probability of the corresponding bin in the word histogram. The higher the probability of the word, the smaller the number of bits of the coded word bci used to encode such word.

The other information that can be extracted from the histogram over the rank-**1** support regards the number of keypoints KP which are included in each cell of the histogram map comprising at least one keypoint KP. Such information is arranged in a corresponding histogram, referred to as histogram count. Each bin of the histogram count corresponds to a corresponding one among the cells of the rank-**1** support that includes at least one keypoint KP. The histogram count lists for each bin the number of keypoints KP included in the corresponding cell. The histogram map of the example at issue is illustrated in FIG. **8**B, wherein 11 cells includes a single keypoint KP each and two cells include two keypoints KP each. The bins of the histogram map of FIG. **8**B are ordered following a column-wise scan of the rank-**1** support.

The keypoint counting information provided by the histogram count is encoded into a set of coded words wj (j=1, 2, . . .) of different lengths, with each coded word wj of the set that indicates which bin(s) of a respective set of histogram count bins correspond to a number of keypoints KP greater than or equal to a certain value.

More specifically, if the highest number of keypoints KP counted within each bin is equal to Nmax, such set of coded words wj comprises a number of coded words wj equal to Nmax−2. The generation of each coded word wj is carried out by performing a corresponding one among a set of Nmax−2 procedure steps. According to an embodiment of the present invention, such procedure steps are described hereinbelow.

Step 1—A first coded word w**1** is set to include an element for each bin of the histogram map. Therefore, the first coded word w**1** includes a number of elements equal to the number bins of the histogram map. Each element of the first coded word w**1** is set to a first value (e.g., "1") if the corresponding bin of the histogram count corresponds to a number of keypoints KP higher than one, otherwise is set to a second value (e.g., "0"). If Nmax is higher than 2, a second step is performed for generating a second coded word w**2**, otherwise the process is termined. In the latter case, the whole information provided by the histogram count results to be coded with the first coded word w**1** only.

Step j (j>1)—A j-th coded word wj is generated. The j-th coded word wj is set to include an element for each bin of the histogram map including more than j keypoints KP. Therefore, the j-th coded word wj includes a number of elements equal to or lower than the j−1 coded word w(j−1). Each element of the j-th coded word wj is set to the first value if the corresponding bin of the histogram count corresponds to a number of keypoints KP higher than j, otherwise is set to the second value. If Nmax is higher than j+1, a (j+1)-th step is performed, for generating a (j+1)-th coded word w(j+1), otherwise the process is termined. In the latter case, the whole information provided by the histogram count is coded with the coded words w**1**−wj.

The compression operations carried out in phase **150** of the extraction procedure **100** (see FIG. **1**) allow to obtain for the coordinates C of the keypoints KP belonging to the subset SUB a corresponding compressed coordinate set CC comprising:

the array r and the array c;
the coded words bci, and
the coded words wj.

The amount of data required for managing (memorizing and/or transmitting) the compressed coordinate set CC is sensibly lower than the amount of data required for managing the set of (uncompressed) coordinates C.

Matching Procedure (FIG. **9**)

FIG. **9** illustrates in terms of functional blocks an image analysis procedure according to an embodiment of the present invention, hereinafter referred to as "matching procedure" and identified with the reference **900**, directed to perform the comparison between two images I**1**, I**2**, by exploiting for each image a respective optimal subset of keypoints and the corresponding compressed descriptors and coordinates generated with the extraction procedure **100** of FIG. **1**.

The steps of the matching procedure **900** may be carried out by proper processing units; for example, each processing unit may be a hardware unit specifically designed to perform one or more steps of the procedure. A possible scenario may provide for a user (client side) which desires to exploit an image comparison service (server side) for comparing the image I**1** with the image I**2**. In this case, the images I**1** and I**2** may be processed at the client according to the extraction procedure **100** of FIG. **1** for the generation of the optimal subset of keypoints and the corresponding compressed descriptors and coordinates; then, the optimal subset of keypoints and the corresponding compressed descriptors and coordinates are sent to the server, which performs the matching procedure **900** exploiting the received data and then provides the results to the client. In this case, the extraction procedure **100** may be carried out by processing units located at the client, e.g., by means of a user's smartphone, while the matching procedure **900** may be carried out by processing units located at the server, e.g., by means of one or more server units adapted to offer image comparison services. Another possible scenario may provide instead that the matching procedure **900** is directly performed at the client. Mixed scenarios are also contemplated, in which the matching procedure **900** is carried out at the client with the compressed descriptors and coordinates sent by the server.

The compressed coordinates of the image I**1** are identified with reference CC**1**, while the compressed descriptors of the image I**1** are identified with reference CDA**1**. Similarly, the compressed coordinates of the image I**2** are identified with reference CC**2**, while the compressed descriptors of the image I**2** are identified with reference CDA**2**.

The compressed descriptors CDA**1** of the first image I**1** are decompressed in order to retrieve corresponding (decompressed) descriptors D**1** (phase **902**). Similarly, the compressed descriptors CDA**2** of the second image I**2** are decompressed in order to retrieve corresponding (decompressed) descriptors D**2** (phase **904**). The decompression of the

descriptors may be carried out by means of reversed versions of the compression operations performed in phase **140** of the extraction procedure **100**. Making reference to descriptors of the SIFT type, after phases **902** and **904** the descriptors D**1** and D**2** are thus represented by corresponding descriptor arrays formed by 128 items.

At phase **906**, matches among descriptors D**1** of the first image I**1** and descriptors D**2** of the second image I**2** are formed by exploiting any one among the feature matching algorithms known in the art, such as for example the Euclidean distance ratio test.

Then, at phase **908**, geometric verification operations are performed for ascertaining which matches among those formed at phase **906** are correct (inliers) and which matches are uncorrected (outliers). As it is known to those skilled in the art, an operation of this type requires, in addition to the descriptors, the coordinates of each keypoint whose corresponding descriptor has been matched with the descriptor of another one keypoint. For this purpose, the compressed coordinates CC**1** of image I**1** and the compressed coordinates CC**2** of the image I**2** should be decompressed as well, for example by means of reversed versions of the compression operations performed in phase **150** of the extraction procedure **100**. The phase dedicated to the decompression of the compressed coordinates CC**1** is identified in FIG. **9** with reference **910**, while the phase dedicated to the decompression of the compressed coordinates CC**2** is identified in FIG. **9** with reference **912**. Once the inliers have been identified, the geometric verification may provide as a result a parameter DOM indicative of the degree of match between image I**1** and I**2**. For example, if such parameter DOM is higher than a predetermined threshold, the images I**1** and I**2** are reputed to depict a same object(s)/scene(s).

Additionally, localization operations (phase **914**) may be further carried out for retrieving the location(s) L of such same object(s)/scene(s) within the two images I**1**, I**2**.

Making reference to the previously mentioned client-server image comparison scenario, since the matching procedure **900** is configured to operate with a reduced number of keypoints (only the ones belonging to the subset SUB generated by means of the extraction procedure **100**), and since the descriptors and the coordinates of such reduced number of keypoints are received in a compressed way, with the proposed solution the overall amount of data to be sent from the client to the server is drastically reduced compared to the known solutions.

Retrieval Procedure (FIG. **10**)

FIG. **10** illustrates in terms of functional blocks an image analysis procedure according to an embodiment of the present invention, hereinafter referred to as "retrieval procedure" and identified with the reference **1000**, in which a query image—such as the query image **115** of FIG. **1**—depicting an object/scene to be recognized is compared with a plurality of model images—each one depicting a respective known object/scene—stored in a model database, in order to retrieve the model image(s) depicting the same object/scene depicted in the query image.

Like the matching procedure **900** of FIG. **9**, the steps of the retrieval procedure **1000** may be carried out by proper processing units; for example, each processing unit may be a hardware unit specifically designed to perform one or more steps of the procedure. A typical scenario may provide for an user (client side) which desires to exploit an image recognition service (server side) in order to automatically recognize an object/scene depicted in a query image **115**. In this case, the query image **115** may be processed at the client according to the extraction procedure **100** of FIG. **1** for the generation of the optimal subset of keypoints SUB and the corresponding compressed descriptors CDA and coordinates CC; then, the optimal subset of keypoints and the corresponding compressed descriptors and coordinates are sent to the server, which performs the retrieval procedure **1000** exploiting the received data and then provides the results to the client. The plurality of model images to be used for the recognition of the object/scene depicted in the query image **115** are stored in a model database **1002**, which is located at server side.

The compressed descriptors CDA are decompressed in order to retrieve corresponding (decompressed) descriptors DD (phase **1004**). The decompression of the descriptors may be carried out by means of reversed versions of the compression operations performed in phase **140** of the extraction procedure **100**. Again, making model to descriptors of the SIFT type, after phase **1004** the descriptors DD are thus represented by corresponding descriptor arrays formed by 128 items.

Since a standard object recognition procedure typically require the execution of comparison operations between the query image and a very high number of model images (for example, the model images included in the model database **1002** may be a few millions), such procedure is both time and memory consuming. For this purpose, a known solution provides for performing such comparison operations in two distinct phases. Instead of directly comparing the descriptors of the query image with the descriptors of all the model images, a fast, rough, comparison is preliminarly made by among visual words extracted from the query image and visual words extracted from the model images; then, the (refined) comparison of the descriptors is carried out only among the descriptors of the query image and the descriptors of a reduced set of model images chosen based on the preliminary comparison. A visual word is an array obtained by performing a vector quantization of a descriptor; in other words, each visual word is a codeword of a visual codebook. The generation of the visual words is carried out for each descriptor of the query image and each descriptor of the model images. For example, the preliminary comparison is carried out by counting the number of visual words in common between the query image and each model image. Then, for each model image, a similitude rank is calculated based on the counts of the number of visual words in common. Similar considerations apply if the similitude rank is generated by comparing the visual words using alternative methods. In this, way, the refined comparison between descriptors may be advantageously carried out only among the query image and the model images having the highest similitude ranks (i.e., the ones having the highest numbers of visual words in common with the query image). This approach, which is derived from the text analysis field, is also known as "ranking by means of Bag of Features (BoF)".

Making reference again to FIG. **10**, in order to allow the carrying out of the ranking by means of BoF, visual words VD for each descriptor of the query image and visual words VDR for each descriptor of each model image have to be generated.

It is pointed out that in order to allow the comparison between visual words, both the visual words VD and the visual words VDR should be generated using a same codebook.

While the visual words VD of the query image **115** have to be generated every time the retrieval procedure **1000** is performed (phase **1006**), in order to drastically reduce the operation times, the generation of the visual words VDR of the model images may be advantageously carried out only once, and then the resulting plurality of visual words VDR may be directly stored in the model database **1002**; alternatively, the visual words VDR may be periodically updated.

Having generated for each descriptor DD of the query image a corresponding visual word VD, in phase **1008** the ranking by means of BoF procedure is then carried out. In this way, for each model image, a rank index is calculated by counting the number of visual words VDR of such model image which are also visual words VD of the query image. Such counting may be carried out using the known ranking by means of BoF implementation also known as Invertedindex. However, similar considerations apply in case different implementations are applied. Once all the rank indexes have been calculated, a list is generated in which the model images of the database are sorted according to a rank index decreasing order. Then, a set SR of model images having the highest rank index values is selected for being subjected to the subsequent (refined) comparison operations.

It is pointed out that since according to an embodiment of the present invention the number of descriptors of each image is advantageously reduced, corresponding only to the optimal subset SUB of keypoints which are considered relevant (see phase **130** of the extraction procedure **100** of FIG. **1**), the amount of data required for carrying out the ranking by means of BoF procedure (phase **1008**) which has to be loaded in the working memory (e.g., in RAM banks located on the server side) is strongly reduced, drastically improving the speed of the process. Moreover, since the comparisons are made by taking into consideration only the descriptors of the keypoints reputed relevant, the precision of the comparison is increased, because the noise is reduced. In order to further improve the speed and the precision, optimal subset including a reduced number of descriptors are also generated for each model image included in the model database **1002**.

It has been found that the number of keypoints forming the optimal subset SUB strongly influence the outcome of the ranking by means of BoF. Indeed, with a same number of considered images, the probability that the object/scene depicted in the query image **115** is also depicted in at least one of the model images belonging to the selected set SR of model images increases as the number of keypoints of the optimal subset SUB decreases. However, if such number of keypoints of the optimal subset SUB falls below a lower threshold, the performances of the procedure decrease, since the number of keypoints included in the subset SUB become too small for satisfactorily representing each image.

At this point, a second, refined comparison is carried out between the query image **115** and the set SR of model images (phase **1010**). One of the already known feature matching procedures may be employed for matching descriptors DD of the query image **115** with descriptors of the model images of the set SR (sub-phase **1012**), e.g., by calculating Euclidean distances among the descriptors, and then a geometric verification is performed for ascertaining which matching are inliers and which are outliers (sub-phase **1014**). In this way, if it exists, the model image RI of the set SR depicting an object/scene depicted also in the query image **115** is retrieved at the end of the phase.

According to an embodiment of the present invention, instead of directly performing feature matching operations on the descriptors DD of the query image **115** and on the descriptors of the model images of the set SR, the feature matching operations are carried out on compressed versions thereof obtained by subdividing the corresponding descriptor arrays into sub arrays and compressing each sub-array by means of a codebook based on vector quantization. For this purpose, the descriptors DD of the query image **115** are compressed at phase **1016**, for example by subdividing the corresponding descriptor arrays in four sub-arrays and compressing each one of said four sub-arrays with a respective codebook. Similarly to the generation of the visual words, the model database **1002** stores for each model image corresponding pre-calculated compressed versions thereof, which have been compressed using the same codebooks used for compressing the descriptors DD of the query image **115**. According to this embodiment, the feature matching (sub-phase **1012**) can be performed in a very fast and efficient way. Indeed, since the feature matching is carried out in the compressed space (both the descriptors of the query image and of the model images are compressed), and since the number of descriptors to be considered is reduced (corresponding only to the keypoints of the optimal subset), it is possible to directly load in the main memory also the data representing the model images of the model database. Moreover, since the compression of the descriptor arrays has been carried out by subdividing the descriptor arrays in sub-arrays, thus strongly reducing the number of codewords of the corresponding codebooks, a list including all the possible Euclidean distances among each codeword of each codebook may by pre-calculated in advance, and loaded in the main memory, further increasing the speed of sub-phase **1012**. Similar considerations apply if the feature matching is carried out by exploiting a different algorithm which does not make use of the Euclidean distances.

According to an embodiment of the present invention, sub-phase **1012** may be further improved by compressing the sub-arrays of each descriptor using a same codebook, using an approach similar to that used in phase **140** of the extraction procedure **100** of FIG. **1**.

Since the geometric verification (sub-phase **1014**) requires, in addition to the descriptors, the coordinates of the keypoints whose corresponding descriptors have been matched with descriptors of another keypoints, the compressed coordinates CC of the keypoints of the query image **115** should be decompressed as well (phase **1018**).

Codebook Compression (FIGS. **11** and **12**)

All the above described solutions provide for compressing the descriptor arrays through vector quantization, by exploiting at least one codebook. As described in detail in the foregoing, a codebook is formed by a plurality of codewords, each approximating a corresponding set of descriptor arrays. Codebooks for compressing descriptor arrays to be used in image analysis procedures are formed by a high amount of data, and therefore entail the occupation of a large amount of working memory. For example, considering descriptors of the SIFT type, a typical codebook for compressing a descriptor array may occupy few hundreds of MB. According to the embodiments of the inventions previously described, the memory occupation of the codebook is advantageously reduced by subdividing the descriptor array into sub-arrays and compressing at least two sub-arrays using a single, same codebook. Thanks to this approach, the memory occupation of the codebook can be reduced to few MBs, such as for example 4-5 MBs. However, in some cases such memory occupation still could be excessive, especially considering when the codebook has to be stored in a unit having a low storage capability, such as for example in a dedicated chip installed on a portable device.

FIG. **11** is a flow chart **1100** illustrating the main phases of a codebook compression procedure according to an embodiment of the present invention.

The procedure is aimed at compressing a codebook CBK formed by N K-tuple codewords CW so that its memory occupation is restricted below a given maximum threshold THC.

The first phase of the procedure (block **1110**) provides for representing the codebook CBK in a matrix form. For this

purpose, a matrix C is generated including a corresponding row for each codeword CW of the codebook CBK, with each row of the matrix C that includes an item for each one of the K elements forming the corresponding codeword CW. Therefore, the matrix C has a N×K size.

The next phase (block **1120**) comprises generating a matrix C' of N×K size from the matrix C by calculating an average 1×K array m obtained from the average of all N rows of the matrix C and then subtracting such average array m from each row of the matrix C. This operation is performed since such average strongly influences the results of the subsequent operations of the procedure.

The next phase (block **1130**) of the procedures provides for eigenvector factorizing the matrix C' in the form $U*S*V^T$, wherein U is a N×K matrix, and S and V are K×K matrices (V^T is the transpose of V), with U and V that are orthonormal matrices, and S is a diagonal matrix, whose diagonal items Sj,j are the singular values of C', and generally decrease as the index j increases. For example, the product $U*S*V^T$ may be obtained from C' by using the known Singular Value Decomposition (SVD) factorization.

With this peculiar type of factorization, the information content of C' is concentrated mainly in the first (i.e., the leftmost) columns of U and V, with the information content corresponding to a generic j-th column CUj of U and to a generic j-th column CVj of V that is higher than the information content corresponding to the (j+1)-th column CU(j+1) of U and to the (j+1)-th column CV(j+1) of V, respectively.

The subsequent phase (block **1140**) provides for approximating the matrix C' with the product $C'm=Um*Sm*Vm^T$, wherein:

- Um is a matrix of size N×K', with K'=K−R, obtained from U by eliminating the last (i.e., the rightmost) R columns thereof;
- Sm is a matrix of size K'×K', obtained from S by eliminating the last (i.e., the rightmost) R columns and the last (i.e., the lowermost) R rows thereof, and
- Vm is a matrix of size K×K', obtained from V by eliminating the last (i.e., the rightmost) R columns thereof.

Since with SVD factorization the information content of C' has been concentrated mainly in the first (i.e., the leftmost) columns CUj, CVj of U and V, approximating C' by eliminating the rightmost columns CUj, CVj of U and V involves the lowest loss of information content for the same number of eliminated columns.

The next phase (block **1150**) comprises selecting a particular compact numeric representation, e.g. through selecting a particular data type (e.g., the int**16** data type or 16 bit signed integer), multiplying each item of Vm by a coefficient SC equal to the ratio between the highest value that can be represented with the selected data type (e.g., 32767 in case of int**16** datatype) and the highest absolute value among the absolute values of the items of Vm, and representing each item (amplified through SC) of Vm using the selected data type, obtaining a corresponding compressed matrix Vm'.

The following phase (block **1160**) provides for generating a N×K' matrix US by multiplying each item of the matrix Um*Sm by the coefficient SC, and then quantizing each item US(i,j) of US into a corresponding quantized item US'(i,j) through scalar quantization taking into account that the information content corresponding to the generic j-th column CUSj of US decreases as j increases. Specifically:

1) each item US(i,**1**) of the first column CUS**1** of US is quantized into a corresponding quantized item US'(i,**1**) using a number Q**1** of different quantization levels QL**1**(y), y=1, . . . , Q**1**;

2) each item US(i,**2**) of the second column CUS**2** of US is quantized into a corresponding quantized item US'(i,**2**) using a number Q**2**≤Q**1** of different quantization levels QL**2**(y), y=1, . . . , Q**2**;

3) each item US(i,**3**) of the third column CUS**3** of US is quantized into a corresponding quantized item US'(i,**3**) using a number Q**3**≤Q**2** of different quantization levels QL**3**(y), y=1, . . . , Q**3**;

. . .

K') each item US(i,K') of the K'-th column CUSK' of US is quantized into a corresponding quantized item US'(i,K') using a number QK'≤Q(K'−1) of different quantization levels QLK'(y), y=1, . . . , QK',

obtaining thus a corresponding compressed N×K' matrix US'.

Then (block **1170**), each possible quantization level QLj (y), j=1 to K', y=1 to Qj, is associated to a respective numeric index INj(y), and a N×K' size index matrix X is generated whose generic item Xi,j is set to the numeric index INj(y) associated to the quantization level QLj(y) assumed by the quantized item US'(i,j) of the matrix US'. For example, the numeric indexes INj(y) may be set in such a way that INj(**1**)=0, INj(**2**)=1, . . . , INj(Qi)=Qi−1, so that the generic item Xi,j of the index matrix X can be represented with $\ln_2 Qj$ bits.

According to an embodiment of the present invention, the items US(i,j) of the matrix US are quantized into the corresponding quantized items US'(i,j) of the matrix US' through non-uniform quantization. Therefore, according to this embodiment of the invention, the Qj quantization levels QLj (y), y=1 to Qj, used to generate the quantized items US'(i,j), i=1 to N, of the j-th column CUS'j of US' are non-uniformly distributed, i.e., with QLj(y)−QLj(y−1) that is in general different than QLj(y+1)−QLj(y). In this case, an index quantization mapping function is generated in form of a table listing for each numeric index INj(y), y=1 to Qj the corresponding associated quantization level QLj(y).

According to a further embodiment of the present invention, in order to simplify the quantization mapping function, the items US(i,j) of the matrix US are quantized into the corresponding quantized items US'(i,j) of the matrix US' through uniform quantization. According to this embodiment of the invention, the Qj quantization levels QLj(y), y=1 to Qj, used to generate the quantized items US'(i,j), i=1 to N, of the j-th column CUS'j of US' are uniformly distributed, i.e., with QLj(y)−QLj(y−1) that is equal to QLj(y+1)−QLj(y). In this case, the index quantization mapping function may be simply formed by a base value equal to QLj(**1**) and a step value equal to (QLj(Qj)−QLj(**1**))/(Qj−1): given a numeric index INj(y), the corresponding quantization level QLj(y) is obtained by multiplying such index INj(y) for the step value and summing the result to the base value.

Therefore, at the end of the last phase of the codebook compression procedure illustrated in FIG. **11**, the codebook CBK may be stored in a memory unit (for example, integrated on a chip installed on a portable device) occupying a memory space MS formed by:

- the memory space MS**1** for storing the average array m of C; the memory space MS**2** for storing the index matrix X;
- the memory space MS**3** for storing the quantization mapping function, and
- the memory space MS**4** for storing the compressed matrix Vm'.

It is underlined that instead of having to directly store the compressed matrix US', according to this embodiment of the present invention the quantized items US'(i,j) of said matrix US' can be advantageously retrieved from the numeric indexes INj(y) forming the items Xi,j of the index matrix X

(with said numeric indexes INj(y) that occupy a lower amount of memory with respect to the quantized items US'(i,j)).

In order to respect the above mentioned memory occupation maximum constraint, MS1+MS2+MS3+MS4 have to be lower than the maximum threshold THC.

On this regard:

MS1 is equal to K multiplied by the memory occupation of the single item of C;

MS2 is equal to N multiplied by the memory occupation MSR of a generic row RXi of X (i.e., the total number of bits required to represent all the items of such row), wherein MSR is equal to the sum of:

1) the memory occupation of the first item Xi,1 of a generic row RXi of X, i.e., the number of bits ($\ln_2$Q1) for representing one of the Q1 numeric indexes IN1 (y), y=1, . . . , Q1;

2) the memory occupation of the second item Xi,2 of a generic row RUSi of US', i.e., the number of bits ($\ln_2$Q2) for representing one of the Q2 numeric indexes IN2(y), y=1, . . . , Q2;

. . .

K') the memory occupation of the K'-th item Xi,K' of a generic row RUSi of US', i.e., the number of bits ($\ln_2$QK') for representing one of the QK' numeric indexes INK'(y), y=1, . . . , QK';

MS3 is given by the type of quantization used for generating the items of US', and

MS4 is given by K×K' multiplied by the data type selected for representing each item of Vm'.

The choice of K' (number of columns of Vm', US' and X) and of the number Q1, Q2, . . . , QK' of quantization levels QLj(y) strongly influences the size of MS2, MS3 and MS4. Since MS2>>MS3+MS4, according to an embodiment of the present invention K' and Q1, Q2, . . . , QK' are calculated by restricting MS2 below a given threshold.

FIG. **12** is a flow chart **1200** illustrating the main phases of a method for selecting the values of K' and Q1, Q2, . . . , QK' according to an embodiment of the present invention.

The first phase (block **1210**) provides for calculating the energy ES of the matrix S. In the present document, with the term "energy of a matrix" it is intended the square of the Frobenius norm of said matrix, while with the term "energy of a matrix item" it is intended the absolute square of such item. As it is well known to those skilled in the art, the Frobenius norm of a matrix is given by the square root of the sum of the absolute squares of its items.

Then (block **1220**), a target memory occupation value THM for MS2 is set. With such target memory occupation value THM, the memory occupation MSR of a generic row RXi of X becomes THM/N. Therefore, THM corresponds to the target total number of bits for representing all the items of X.

The next phase (block **1230**) provides for calculating, for each diagonal item Sj,j (j=1, 2, . . . , K) of the matrix S, a corresponding allocation value Pj equal to the target memory occupation value THM/N (rounded to integer) multiplied by a ratio between the energy Ej of such item (i.e., the square thereof) and the total energy ES. The ratio Ej/ES is proportional to the information content corresponding to the j-th column CUSj of US. Since Sj,j decreases as j increases, also Ej/ES decreases as j increases. The value Pj corresponding to the item Sj,j provides an indication about the fraction of the target memory occupation value THM which will be allocated to represent the items Xi,j of the j-th column CXj of X. The higher the value of j, the lower the allocation value Pj, since the lower the information content corresponding to the j-th column CUSj of US.

Each allocation value Pj is then rounded to an integer value PIj (block **1240**). This rounded allocation value PIj is the number of bits assigned to represent the generic item Xi,j belonging to the j-th column CX'j of X.

At block **1250**, K' (i.e., the number of columns of Vm, Vm', US, US', and X) is set to the highest j for which PIj is higher than a threshold, preferably equal to zero.

Then (block **1260**), for j=1 to K', the number of quantization levels Qj used for quantizing the generic item USi,j belonging to the j-th column CUSj of US so as to generate the corresponding item US'i,j of US' is set to 2^(PIj). The higher the rounded allocation value PIj, the higher the number of quantization levels Qj.

Therefore, according to the method described above, each item of US is advantageously quantized using a number Qj of quantization levels QLj(y) based on the information content of the column CUSj which belongs to.

Once the values of K' and Q1, Q2, . . . , QK' are determined with the method described above, the memory space MS3 for storing the quantization mapping function depends of the type of quantization used to quantize the items US(i,j) of the matrix US into the corresponding quantized items US'(i,j) of the matrix US'.

In case of non-uniform quantization, MS3 is equal to the memory occupation of a data table including for each column of CUSj of US a number Qj of data table rows, each one listing a numeric index INj(y), y=1 to Qj, and the corresponding associated quantization level QLj(y).

In case of uniform quantization, MS3 is equal to, for each column of CUSj of US, the memory occupation of a base value (e.g., equal to QLj(1)) and a step value (e.g., equal to (QLj(Qj)−QLj(1))/(Qj−1)).

According to an embodiment of the present invention, instead of factorizing the matrix C' in the form $U*S*V^T$ by directly applying the SVD on the matrix C' (a very heavy computational burden in case C' is large), the SVD is advantageously applied to the smaller square matrix C'^T*C', sensibly speeding up the SVD computation. By applying the SVD to the square matrix C'^T*C', such matrix is factorized in the form $V*S^2*V^T$. By applying the method previously described with reference to FIG. **12** to S, easily obtainable from S^2 with a simple operation, the values for K' and Q1, Q2, . . . , QK' are selected. The matrix Vm is obtained from V by eliminating the last R=K−K' columns from V. Then, the compressed matrix Vm' is generated from Vm as in the procedure described with reference to FIG. **11** (see block **1150**). At this point, the matrix US is obtained by multiplying C' by Vm, the items of US are quantized and the index matrix X is generated as in the procedure described with reference to FIG. **11** (see blocks **1160** and **1170**).

The procedures according to the embodiments of the invention described with reference to FIGS. **11** and **12** can be also applied to compress the so-called tree-structured codebooks. A tree-structured codebook is a codebook arranged in a plurality of levels, wherein each codeword of a level is associated to a corresponding set of codewords in the subsequent level. When a descriptor array is quantized using a tree-structured codebook, a first, rough quantization is carried out using the first level of the codebook, selecting the codeword of such first level that better approximate the descriptor array. Then, a second quantization is carried out using the second level of the codebook, by inspecting only the codewords thereof corresponding to the selected codeword of the first level. The process is reiterated until reaching the last level of the tree-structured codebook.

According to an embodiment of the present invention, a tree-structured codebook including x levels may be com-

pressed with the procedure previously described with the matrix C formed in the following way:

- a first group of rows of C are formed by the codewords of the first level of the tree-structured codebook;
- a second group of rows of C are formed by the codewords of the second level of the tree-structured codebook;
- . . .
- an x-th group of rows of C are formed by the codewords of the x-th level of the tree-structured codebook.

Then, the compression procedure is carried out as already described for the single-level codebook, until the determining of the values for K' and Q1, Q2, . . . , QK'. At this point, using such values K' and Q1, Q2, . . . , QK', the codebook of each level is individually compressed by applying the compression operations already described for the single-level codebook to an US matrix obtained by considering only the codewords of said level.

It is underlined that although the codebook compression procedures described in detail in the present description have been directed to compress portions of a descriptor array (sub-array), the concepts of the present invention can be directly applied to a full descriptor array.

Codebook Compression—Exemplary Matrix Values

This section of the description provides a numeric example of some of the matrices generated during the procedure described with reference to FIG. **11**.

C = CBK

C =

9.2112	8.6746	10.0395	11.4650	13.5815	11.2922	8.9053	8.4101
138.6537	24.1106	1.9616	2.4629	5.3038	2.6119	2.0670	24.3525
49.8938	20.3497	8.5559	8.3388	9.8446	8.3333	7.7843	18.0909
100.6994	27.4951	9.0531	12.8342	19.8983	12.3285	8.4224	25.3448
123.5751	100.4345	18.7715	6.8645	6.0713	5.0030	5.1476	21.1944
125.2223	21.6952	5.2084	5.0772	6.1797	6.9548	17.5039	99.9505
27.8627	9.4863	5.2260	24.9820	109.1541	24.7491	5.0948	9.3586
42.2997	96.9057	32.3036	8.8458	8.3667	8.4285	6.6172	8.6094
48.2478	9.7809	6.9946	8.5767	8.6187	7.8719	22.7389	87.6189
19.0383	19.0822	64.1920	95.1155	22.5266	7.1767	7.1317	10.5471
21.8533	64.5315	105.0803	26.7304	7.9565	6.8195	7.5896	8.3965
17.0069	9.5018	9.7584	8.5147	10.5827	33.9943	91.4393	37.0260
19.8879	10.2558	6.6847	7.1057	24.8338	103.1141	60.0129	18.4336
20.9942	11.1982	17.5985	100.9568	95.0604	13.9414	6.0197	9.7194
20.9369	9.7350	6.0820	14.0420	97.7723	100.6393	17.6851	11.4594
34.0949	7.6463	5.3601	5.6678	6.3209	16.2194	95.6535	112.4096

C' = C − m

m =

51.2174	28.1802	19.5544	21.7238	28.2545	23.0924	23.1133	31.9326

C' =

−42.0062	−19.5056	−9.5149	−10.2587	−14.6730	−11.8002	−14.2080	−23.5225
87.4363	−4.0696	−17.5928	−19.2609	−22.9507	−20.4804	−21.0463	−7.5801
−1.3236	−7.8305	−10.9985	−13.3849	−18.4098	−14.7590	−15.3290	−13.8417
49.4820	−0.6851	−10.5013	−8.8896	−8.3562	−10.7639	−14.6910	−6.5878
72.3577	72.2543	−0.7828	−14.8592	−22.1832	−18.0894	−17.9658	−10.7382
74.0049	−6.4850	−14.3460	−16.6465	−22.0748	−16.1376	−5.6094	68.0179
−23.3547	−18.6940	−14.3284	3.2583	80.8996	1.6568	−18.0185	−22.5740
−8.9177	68.7255	12.7492	−12.8779	−19.8878	−14.6638	−16.4961	−23.3232
−2.9696	−18.3993	−12.5598	−13.1470	−19.6358	−15.2205	−0.3745	55.6863
−32.1791	−9.0980	44.6376	73.3917	−5.7279	−15.9157	−15.9816	−21.3855
−29.3641	36.3513	85.5259	5.0066	−20.2980	−16.2729	−15.5238	−23.5361
−34.2105	−18.6784	−9.7960	−13.2091	−17.6718	10.9019	68.3260	5.0934
−31.3295	−17.9244	−12.8696	−14.6180	−3.4207	80.0218	36.8996	−13.4991
−30.2231	−16.9820	−1.9559	79.2330	66.8059	−9.1510	−17.0936	−22.2132
−30.2804	−18.4452	−13.4724	−7.6818	69.5178	77.5469	−5.4282	−20.4732
−17.1225	−20.5339	−14.1943	−16.0559	−21.9336	−6.8729	72.5401	80.4770

$C' = U*S*V^T$

U =

−0.1135	0.0112	0.1155	−0.0358	0.6114	0.1768	0.0503	0.1696	−0.2587
−0.2639	−0.1185	−0.2691	0.1372	0.1292	0.5083	−0.1382		
0.3642	0.0999	−0.3055	0.0111	0.2483	−0.3015	−0.0449	−0.2929	0.3966
−0.1923	0.2125	−0.1373	0.1174	−0.1832	0.2573	0.3896		
0.0527	0.0393	0.0056	−0.0298	0.4249	0.0680	0.0145	0.0822	−0.2189
0.4816	0.6622	0.1940	−0.0372	0.1044	−0.1453	0.1152		
0.1917	0.0771	−0.1979	0.0065	0.1207	−0.1556	−0.0328	−0.1502	0.0674
0.3904	−0.4106	−0.1262	−0.0995	0.7095	−0.0259	0.0158		
0.3771	0.2863	−0.0984	−0.2477	−0.3180	0.0572	−0.3231	0.1115	−0.0632
0.0455	0.2870	−0.2567	−0.1495	0.0181	0.2665	−0.4853		
0.4082	−0.1639	−0.1745	0.2566	−0.2196	−0.0720	0.3141	0.1016	−0.3215
0.0764	−0.0392	0.3110	0.5246	0.0196	0.2357	−0.0956		
−0.2926	0.0660	−0.3541	0.0978	0.0060	0.4584	0.0185	−0.3930	0.1374
0.3168	−0.0050	−0.2501	0.3797	−0.1884	−0.0300	−0.2184		
0.0917	0.2709	0.2068	−0.3153	−0.1076	0.3806	−0.2015	0.3408	0.2296
0.1774	−0.1783	0.1479	0.3386	0.0166	0.1189	0.4425		
0.1318	−0.2210	0.0782	0.2340	0.0744	0.2220	0.3229	0.3368	0.6840
0.0039	0.1388	0.0198	−0.1111	0.1389	0.0327	−0.2831		
−0.2108	0.2749	0.2867	0.3499	−0.0189	−0.4533	0.0039	0.1672	0.0907

-continued

0.4768	−0.1040	−0.1417	0.0048	−0.3008	0.2858	−0.0529		
−0.0494	0.3448	0.4904	−0.1310	−0.1867	−0.0049	0.4666	−0.4457	0.0408
−0.1179	0.2279	−0.0252	0.1270	0.2811	0.0799	−0.0118		
−0.0862	−0.3408	0.2467	−0.1174	0.0872	−0.0768	−0.4392	−0.3648	0.1908
0.0839	−0.0084	0.4988	0.0804	0.0488	0.3106	−0.2540		
−0.2194	−0.3003	−0.0173	−0.4494	−0.0393	−0.4360	0.0163	0.2278	0.1183
0.0201	0.1520	−0.2942	0.4761	0.1269	−0.1899	−0.1132		
−0.3705	0.2086	−0.1731	0.4626	−0.1838	−0.0400	−0.3409	0.1468	0.0418
−0.2902	0.2890	0.0864	0.1766	0.4257	0.0584	0.0966		
−0.3795	−0.1151	−0.3861	−0.3041	−0.2366	−0.0051	0.3151	0.0696	−0.0012
0.1278	0.0780	0.1518	−0.3204	0.0498	0.5052	0.1873		
0.1044	−0.5379	0.2769	0.2118	−0.2624	0.1822	−0.1398	−0.1072	−0.1202
0.1413	0.1525	−0.4701	−0.0648	0.0889	0.1779	0.3502		

S =

220.4621	0	0	0	0	0	0	0
0	184.9380	0	0	0	0	0	0
0	0	145.8051	0	0	0	0	0
0	0	0	124.4790	0	0	0	0
0	0	0	0	78.3028	0	0	0
0	0	0	0	0	71.4800	0	0
0	0	0	0	0	0	62.0052	0
0	0	0	0	0	0	0	43.7004
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

V =

0.6718	0.1395	−0.5113	0.0483	−0.2001	−0.3954	−0.0574	−0.2565
0.2311	0.3981	0.1907	−0.4079	−0.5292	0.3593	−0.3020	0.2916
−0.0725	0.3355	0.4785	0.0008	−0.3194	−0.2202	0.5149	−0.4874
−0.2904	0.2795	0.0768	0.5619	−0.2610	−0.4683	−0.3318	0.3441
−0.4702	0.0753	−0.5609	0.1572	−0.3948	0.3968	0.0018	−0.3472
−0.3053	−0.2785	−0.2385	−0.5449	−0.3045	−0.4620	0.2691	0.3113
−0.0404	−0.5401	0.2912	−0.0969	−0.2620	−0.1434	−0.5699	−0.4456
0.2995	−0.5064	0.1056	0.4296	−0.4388	0.2351	0.3648	0.2779

Vm = SC*V, considering only the first K' columns

K' = 6

SC = 4.8772e+004

Vm = 1.0e+004 *

−3.2767	0.6805	2.4936	−0.2356	0.9759	−1.9284	0.2798	1.2509
−1.1272	1.9417	−0.9299	1.9896	2.5810	1.7523	1.4729	−1.4221
0.3534	1.6365	−2.3336	−0.0038	1.5577	−1.0740	−2.5115	2.3773
1.4162	1.3634	−0.3747	−2.7405	1.2731	−2.2843	1.6180	−1.6780
2.2933	0.3675	2.7355	−0.7667	1.9258	1.9355	−0.0087	1.6934
1.4890	−1.3582	1.1633	2.6574	1.4852	−2.2533	−1.3126	−1.5181
0.1972	−2.6340	−1.4204	0.4724	1.2778	−0.6993	2.7794	2.1733
−1.4609	−2.4698	−0.5150	−2.0954	2.1399	1.1465	−1.7794	−1.3555

US = C' *Vm

US = 1.0e+006 *

1.2208	0.1014	−0.8213	0.2171	−2.3349	0.6163
−3.9161	0.9009	2.1728	−0.0673	−0.9483	−1.0510
−0.5668	0.3549	−0.0396	0.1811	−1.6226	0.2369
−2.0613	0.6955	1.4074	−0.0395	−0.4610	−0.5424
−4.0552	2.5826	0.6996	1.5039	1.2144	0.1994
−4.3895	−1.4781	1.2406	−1.5580	0.8385	−0.2510
3.1457	0.5950	2.5181	−0.5940	−0.0229	1.5979
−0.9860	2.4434	−1.4709	1.9144	0.4108	1.3269
−1.4171	−1.9932	−0.5563	−1.4208	−0.2842	0.7740
2.2667	2.4797	−2.0392	−2.1246	0.0723	−1.5804
0.5310	3.1105	−3.4877	0.7950	0.7130	−0.0171
0.9272	−3.0744	−1.7546	0.7126	−0.3331	−0.2677
2.3592	−2.7091	0.1233	2.7282	0.1503	−1.5199
3.9835	1.8815	1.2312	−2.8086	0.7019	−0.1396
4.0810	−1.0386	2.7459	1.8465	0.9038	−0.0177
−1.1230	−4.8520	−1.9694	−1.2858	1.0020	0.6353

-continued

Vm'

Vm' =

−32767	6805	24936	−2356	9759	−19284	2798	12509
−11272	19417	−9299	19896	25810	17523	14729	
−14221							
3534	16365	−23336	−38	15577	−10740	−25115	
23773							
14162	13634	−3747	−27405	12731	−22843	16180	
−16780							
22933	3675	27355	−7667	19258	19355	−87	
16934							
14890	−13582	11633	26574	14852	−22533	−13126	
−15181							
1972	−26340	−14204	4724	12778	−6993	27794	
21733							
−14609	−24698	−5150	−20954	21399	11465	−17794	
−13555							

Uniform quantization levels QLj (y)

bitsPerComponent' = 6 5 3 2 1 1

$Levels_1$ = 1.0e+006 *

Columns 1 through 11

−4.3895	−4.2551	−4.1206	−3.9862	−3.8517	−3.7173	−3.5828	−3.4484	−3.3139
−3.1795	−3.0450							

Columns 12 through 22

−2.9105	−2.7761	−2.6416	−2.5072	−2.3727	−2.2383	−2.1038	−1.9694	−1.8349
−1.7005	−1.5660							

Columns 23 through 33

−1.4316	−1.2971	−1.1627	−1.0282	−0.8937	−0.7593	−0.6248	−0.4904	−0.3559
−0.2215	−0.0870							

Columns 34 through 44

0.0474	0.1819	0.3163	0.4508	0.5852	0.7197	0.8541	0.9886	1.1231
1.2575	1.3920							

Columns 45 through 55

1.5264	1.6609	1.7953	1.9298	2.0642	2.1987	2.3331	2.4676	2.6020
2.7365	2.8709							

Columns 56 through 64

3.0054	3.1399	3.2743	3.4088	3.5432	3.6777	3.8121	3.9466	4.0810

$Levels_2$ = 1.0e+006 *

Columns 1 through 11

−4.8520	−4.5952	−4.3383	−4.0814	−3.8246	−3.5677	−3.3109	−3.0540	−2.7972
−2.5403	−2.2835							

Columns 12 through 22

−2.0266	−1.7697	−1.5129	−1.2560	−0.9992	−0.7423	−0.4855	−0.2286	0.0282
0.2851	0.5420							

Columns 23 through 32

0.7988	1.0557	1.3125	1.5694	1.8262	2.0831	2.3399	2.5968	2.8537
3.1105								

$Levels_3$ = 1.0e+006 *

−3.4877	−2.6134	−1.7392	−0.8650	0.0093	0.8835	1.7578	2.6320

$Levels_4$ = 1.0e+006 *

−1.8396	−0.5603	0.7190	1.9982	

$Levels_5$ = 1.0e+005 *

−9.9735	5.9841

$Levels_6$ = 1.0e+005 *

−5.9854	7.6955

US'

US' = 1.0e+006 *

1.2575	0.0282	−0.8650	0.7190	−0.9974	0.7696
−3.8517	0.7988	1.7578	−0.5603	−0.9974	−0.5985
−0.6248	0.2851	0.0093	0.7190	−0.9974	0.7696
−2.1038	0.7988	1.7578	−0.5603	−0.9974	−0.5985
−4.1206	2.5968	0.8835	1.9982	0.5984	0.7696
−4.3895	−1.5129	0.8835	−1.8396	0.5984	−0.5985
3.1399	0.5420	2.6320	−0.5603	0.5984	0.7696
−1.0282	2.3399	−1.7392	1.9982	0.5984	0.7696
−1.4316	−2.0266	−0.8650	−1.8396	−0.9974	0.7696
2.3331	2.5968	−1.7392	−1.8396	0.5984	−0.5985
0.5852	3.1105	−3.4877	0.7190	0.5984	−0.5985
0.9886	−3.0540	−1.7392	0.7190	−0.9974	−0.5985
2.3331	−2.7972	0.0093	1.9982	0.5984	−0.5985
3.9466	1.8262	0.8835	−1.8396	0.5984	−0.5985
4.0810	−0.9992	2.6320	1.9982	0.5984	−0.5985
−1.1627	−4.8520	−1.7392	−1.8396	0.5984	0.7696

-continued

X					
X =					
43	20	4	3	1	2
5	23	7	2	1	1
29	21	5	3	1	2
18	23	7	2	1	1
3	30	6	4	2	2
1	14	6	1	2	1
57	22	8	2	2	2
26	29	3	4	2	2
23	12	4	1	1	2
51	30	3	1	2	1
38	32	1	3	2	1
41	8	3	3	1	1
51	9	5	4	2	1
63	27	6	1	2	1
64	16	8	4	2	1
25	1	3	1	2	2

Codebook Compression—Exemplary Results

This section of the description provides experimental results obtainable by employing a codebook compressed with the method described with reference to FIGS. **11** and **12**.

1) FIRST EXAMPLE

length of the information request ("query"): 512 byte;
descriptor array DA subdivided into two sub-arrays, each one using a codebook CBK including 2^14 codewords CW;
each codeword CW includes 64 unsigned char (1 byte) elements.

1.1) Uncompressed Codebook
Memory occupation equal to 2048 KBs.

1.2) Compressed Codebook
MSR=28;
K'=15
25 bits: 4 3 3 2 2 2 1 1 1 1 1 1 1 1 1
MS**1**: 0.25 KBs
MS**2**: 50 KBs
MS**3**: 0.11719 KBs
MS**4**: 1.875 KBs.
Memory occupation MS=MS**1**+MS**2**+MS**3**+MS**4**=427968 bits=52.2422 KBs

1.3) Compressed Codebook/Uncompressed Codebook Ratio
52.2422/2048=0.0255≅2.5%

2) SECOND EXAMPLE

length of the information request ("query"): 2048 byte;
descriptor array DA subdivided into four sub-arrays, each one using a codebook CBK including 2^14 codewords CW;
each codeword CW includes 32 unsigned char (1 byte) elements.

2.1) Uncompressed Codebook
Memory occupation equal to 2048 KBs.

2.2) Compressed Codebook
MSR=20;
K'=11
25 bits: 4 3 3 2 2 1 1 1 1 1 1
MS**1**: 0.125 KBs
MS**2**: 40 KBs
MS**3**: 0.085938 KBs
MS**4**: 0.6875 KBs.
Memory occupation MS=MS**1**+MS**2**+MS**3**+MS**4**=335040 bits=40.8984 KBs

2.3) Compressed Codebook/Uncompressed Codebook Ratio
40.8984/2048=0.02≅2%

3) THIRD EXAMPLE

length of the information request ("query"): 8192 byte;
descriptor array DA subdivided into eight sub-arrays;
tree-structured codebooks with two levels;
the first level includes 2^7 codewords CW;
the second level includes 2^14 codewords CW;
each one using a codebook CBK including 2^14 codewords CW;
each codeword CW includes 16 unsigned char (1 byte) elements.

2.1) Uncompressed Codebook
Memory occupation equal to 2064 KBs.

2.2) Compressed Codebook
MSR=17;
K'=9;
17 bits: 5 3 2 2 1 1 1 1 1
MS**1**: 0.0625 KBs
MS**2**: 0.26563+34 KBs
MS**3**: 0.0070313 KBs
MS**4**: 0.28125 KBs.
Memory occupation MS=MS**1**+MS**2**+MS**3**+MS**4**=284096 bits=346797 KBs

2.3) Compressed Codebook/Uncompressed Codebook Ratio
346797/2064=0.0168≅1.6%

Uses of the Compressed Codebook

In this last section of the document, few examples are disclosed of how a codebook compressed according to the procedure previously described can be used for carrying out image analysis operation according to an embodiment of the present invention.

Firstly, the compressed codebook may be used to compress descriptor arrays DA, o portions thereof (i.e., the sub-arrays SDAk, see the extraction procedure section of this document). Making reference for example to the case of a 1×K sub-array, the average array m (stored in the memory space MS**1**) is initially subtracted from said sub-array, then the averaged sub-array is multiplied by the compressed matrix Vm' (stored in the memory space MS**4**), in such a way that all the following operations are carried out in the reduced K'-space instead of the K-space. At this point, the Euclidean distances between such sub-array and all the codewords of the

compressed codebook are calculated, and then the codeword having the lower Euclidean distance is selected for approximating the averaged sub-array. Using a compressed codebook according to the embodiments of the invention, the codewords to be used for calculating the Euclidean distances, as well as the codeword selected for approximating the sub-array, would be the rows of the matrix US'. Expediently, instead of having to directly store the matrix US', which would occupy a sensible amount of memory space, according to the embodiments of the present invention the matrix US' is not memorized, but instead the rows thereof, i.e., the codewords forming the compressed codebook, are indirectly retrieved using the indexes of the corresponding rows of the index matrix X (stored in the memory space MS**2**), and then applying thereof the quantization mapping function (stored in the memory space MS**3**).

The opposite operation, i.e., the retrieval of the descriptor array (or sub-array) in the original K-space starting from the indexes of a selected row RXi of the matrix X which identifies a corresponding codeword in the K'-space, provides for the following phases. Firstly, for each element Xi,j RXi of X, j=1 to K', exploiting the quantization mapping function a corresponding quantization level QLj(y) is retrieved, and then a 1×K' array H is formed whose items are the retrieved quantization levels QLj(y). The descriptor array in the K-space is retrieved by multiplying each item of H by a scalar equal to 1/SC^2 (wherein SC is the coefficient used to generate the matrix Vm', see block **1150** of FIG. **11**), multiplying the resulting scaled array H by the transpose of Vm' so as to return to the K-space, and then summing the average array m.

In order to compare two descriptor arrays compressed in the K'-space, it is not necessary to come back to the original K-space. If the descriptor arrays were compressed with the same scheme (e.g., same subdivision in sub-arrays and same codebook compression), the comparison may be indeed carried out by calculating the distances among the respective H arrays, allowing thus to reduce the computational load, since the descriptor array comparison procedure instead of requiring the use of absolute distances, operates with distance ratios comparisons.

Performances of Compressed Codebooks in Exemplary Applications

FIGS. **13**A-**13**G illustrates the performance of image comparison systems based on non compressed codebooks (continuous lines) and based on codebook compressed according to the method described with reference to FIGS. **11** and **12** (dashed lines), with the x axis depicting the length information request ("query") in KBs, and the y axis depicting the success percentage of a correct image matching, calculated by fixing a maximum error on the false positives equal to 1%.

FIG. **13**A refers to the MPEG CDVS image category "graphics";

FIG. **13**B refers to the MPEG CDVS image category "graphics" in the VGA resolution;

FIG. **13**C refers to the MPEG CDVS image category "graphics" in the VGA resolution with the images compressed with a high JPEG compression;

FIG. **13**D refers to the MPEG CDVS image category "paintings";

FIG. **13**E refers to the MPEG CDVS image category "videoframes";

FIG. **13**F refers to the MPEG CDVS image category "buildings";

FIG. **13**G refers to the MPEG CDVS image category "common objects".

The previous description presents and discusses in detail several embodiments of the present invention; nevertheless, several changes to the described embodiments, as well as different invention embodiments are possible, without departing from the scope defined by the appended claims.

The invention claimed is:

1. A method for processing an image, comprising:

identifying a group of keypoints in the image;

for each keypoint of the group

a) calculating a corresponding descriptor array including a plurality of array elements, each array element storing values taken by a corresponding color gradient histogram of a respective sub-region of the image in the neighborhood of the keypoint;

b) generating at least one compressed descriptor array by compressing at least one portion of the descriptor array by means of vector quantization using a codebook comprising a plurality of codewords;

exploiting said at least one compressed descriptor array of the keypoints of said group for analysing the image, wherein the method further comprises:

compressing the codebook, said compressing the codebook including:

generating a codebook matrix, each row of the codebook matrix being a codeword of the codebook;

factorising the codebook matrix so as to obtain the product of at least a first matrix and a second matrix, the energy of the items of the second matrix generally non-increasing as the column indexes of such matrix increase,

truncating the first matrix by removing therefrom a first number of last columns;

truncating the second matrix by removing therefrom the first number of last columns and the first number of last rows;

generating a first further matrix corresponding to the product of the truncated first matrix by the truncated second matrix;

quantizing each item of the first further matrix, wherein each item belonging to a column of the first further matrix is quantized using a corresponding number of quantization levels that is lower than or equal to the number of quantization levels used to quantize the items belonging to a preceding column;

generating a second further matrix wherein each item of the second further matrix corresponds to an item of the first further matrix, each item of said second further matrix being an index associated to the quantization level assumed by the corresponding quantized item of the first further matrix;

storing the codebook by memorizing said indexes of the second further matrix in a memory unit.

2. The method according to claim **1**, wherein the first matrix is an orthonormal matrix and the second matrix is a diagonal matrix whose diagonal items correspond to the singular values of the codebook matrix, each diagonal item belonging to a column of the second matrix, except a last column, being higher than the item belonging to the following column.

3. The method according to claim **1**, wherein said product of at least the first matrix and the second matrix further comprise the transpose of a third matrix, the third matrix being an orthonormal matrix

and wherein said method further comprises:

truncating said third matrix by removing therefrom the first number of last columns;

storing the truncated third matrix in the memory unit.

4. The method according to claim **3**, wherein said truncating the first, the second and the third matrices further comprises:

calculating the energy of the second matrix;

setting a first target value for the memory space occupied by all the quantized items of the second further matrix when memorized in the memory unit;

for each diagonal item of the second matrix, calculating a respective allocation value corresponding to the first target value multiplied by a ratio between the energy of such item and the energy of the second matrix;

rounding each allocation value to an integer value, and

setting the first number to the number of columns of the codebook matrix minus the higher column index of the second matrix for which the rounded allocation value corresponding to the diagonal item belonging to the column of the second matrix identified by such column index is higher than a threshold.

5. The method according to claim **4**, wherein said quantizing each item of the first further matrix further comprises:

for each column of the first further matrix, setting the corresponding number of quantization levels for quantizing each item of said column of the first further matrix to two raised to the power of the rounded allocation value corresponding to the diagonal item belonging to the column of the second matrix having the same column index as said column of the first further matrix.

6. The method of claim **3**, further including:

selecting a data type;

scaling each item of the truncated third matrix by a coefficient corresponding to the ratio between the highest value that can be represented with the selected data type and the highest absolute value among the absolute values of the items of the truncated third matrix, and

representing each scaled item of the truncated third matrix with the selected data type, said storing the codebook further including memorizing the scaled items of the tuncated third matrix represented with the selected data type in the memory unit.

7. The method of claim **6**, wherein said generating the first further matrix comprises multiplying the truncated first matrix by the truncated second matrix and then multiplying each item of the resulting matrix by the coefficient.

8. The method of claim **1**, wherein said factorizing the codebook matrix comprises using the Singular Value Decomposition.

9. The method of claim **1**, further including:

before factorising the codebook matrix into the product of at least the first and second matrices, calculating an average array obtained from the average of all the rows of the codebook matrix and then subtracting such average array from each row of the codebook matrix, said storing the codebook further including memorizing the average array in the memory unit.

10. A system for processing an image comprising a group of keypoints, said system being configured to perform the following operations:

for each keypoint of the group:

a) calculating a corresponding descriptor array including a plurality of array elements, each array element storing values taken by a corresponding color gradient histogram of a respective sub-region of the image in the neighborhood of the keypoint;

b) generating at least one compressed descriptor array by compressing at least one portion of the descriptor array by means of vector quantization using a codebook comprising a plurality of codewords;

exploiting said at least one compressed descriptor array of the keypoints of said group for analysing the image, wherein the method further comprises:

compressing the codebook, said compressing the codebook including:

generating a codebook matrix, each row of the codebook matrix being a codeword of the codebook;

factorising the codebook matrix so as to obtain the product of at least a first matrix and a second matrix, the energy of the items of the second matrix generally non-increasing as the column indexes of such matrixes increase,

truncating the first matrix by removing therefrom a first number of last columns;

truncating the second matrix by removing therefrom the first number of last columns and the first number of last rows;

generating a first further matrix corresponding to the product of the truncated first matrix by the truncated second matrix;

quantizing each item of the first further matrix, wherein each item belonging to a column of the first further matrix is quantized using a corresponding number of quantization levels that is lower than or equal to the number of quantization levels used to quantize the items belonging to a preceding column:

generating a second further matrix wherein each item of the second further matrix corresponds to an item of the first further matrix, each item of said second further matrix being an index associated to the quantization level assumed by the corresponding quantized item of the first further matrix;

storing the codebook by memorizing said indexes of the second further matrix in a memory unit.

* * * * *